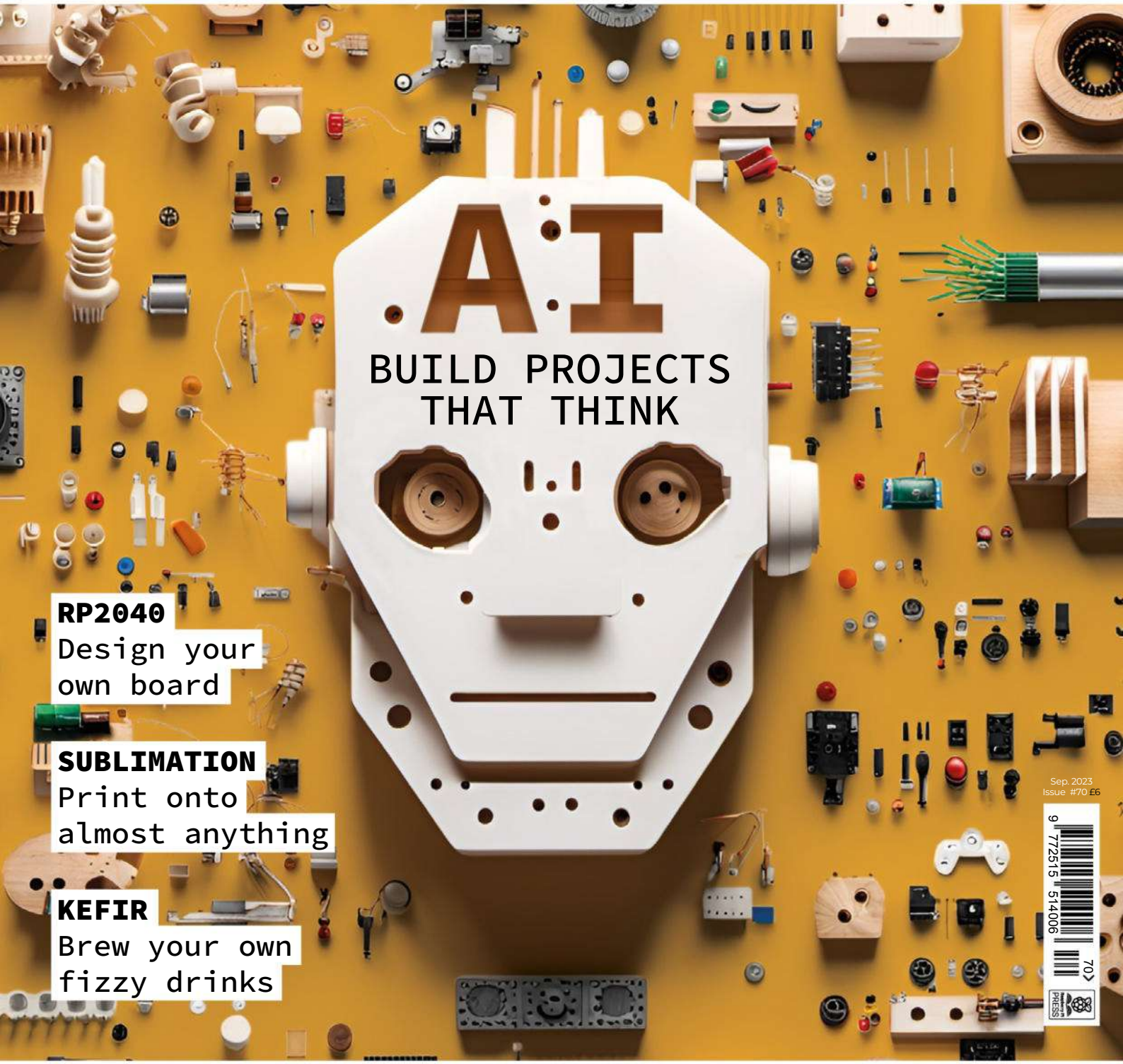
MAKE | BUILD | HACK | CREATE

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.CC | September 2023 | Issue #70



TOMBOLA **BAT** CAMERA **INTEGZA**

Free eBook!

Download your copy from

hsmag.cc/freecadbook

Welcome to HackSpace magazine

In the last year or so, AI has been pushing forwards at almost every level. Neural networks have been squeezed down to fit on small microcontrollers to allow them to sift data for interesting anomalies. At the same time, data centres are being loaded up with specialist hardware to craft realistic images (such as the one on this issue's cover).

AI is a constantly evolving topic that has long been the preserve of universities. It has now broken out of academia and is living amongst us. It's possible to train systems on your own data to think in the ways you want them to, tap into language systems so advanced they almost seem sentient, or pluck fully-formed images from a silicon brain.

Some people claim that this AI will be great for the future of humanity; others are less positive. Here at HackSpace magazine, we're not quite so passive. It's a technology: it can be used in ways that are beneficial, and ways that are damaging. The more creators that have the ability to work with it and create their own intelligences, the more of the potential benefits we can realise. Let's not accept the future that's given to us – let's build the one we want.

BEN EVERARD
Editor ben.everard@raspberrypi.com

Got a comment, question, or thought about HackSpace magazine?

get in touch at **hsmag.cc/hello**

GET IN TOUCH

hackspace@raspberrypi.com
hackspacemag
hackspacemag

ONLINE

hsmag.cc

Available on the App Store

GET IT ON Google Play

EDITORIAL

Editor
Ben Everard
ben.everard@raspberrypi.com

Features Editor
Andrew Gregory
andrew.gregory@raspberrypi.com

Sub-Editors
David Higgs, Nicola King

DESIGN

Critical Media and Raspberry Pi
criticalmedia.co.uk

Head of Design
Lee Allen

Designers
Sam Ribbits, Sara Parodi, Jack Willis

Photography
Brian O'Halloran

CONTRIBUTORS

Marc de Vinck, Andrew Lewis, Jo Hinchliffe, Brendan Charles, Phil King, Rob Miles

PUBLISHING

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Advertising
Charlie Milligan
charlotte.milligan@raspberrypi.com

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6, The Enterprise Centre,
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE

To subscribe
01293 312189
hsmag.cc/subscribe

Subscription queries
hackspace@subscriptionhelpline.co.uk

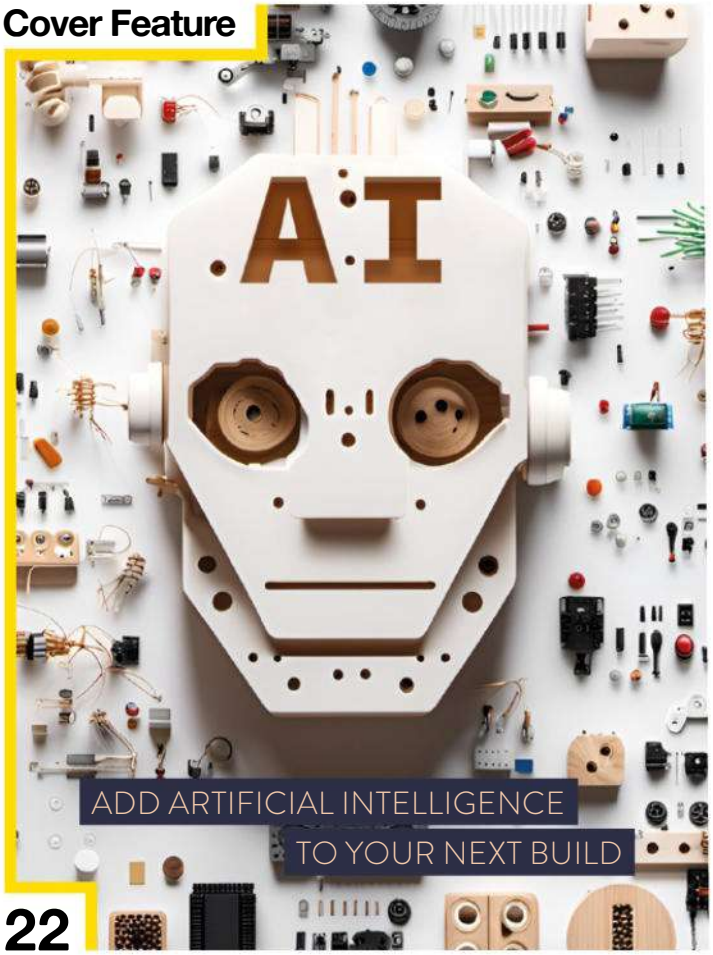
Contents

06

06 SPARK

21 LENS

Cover Feature



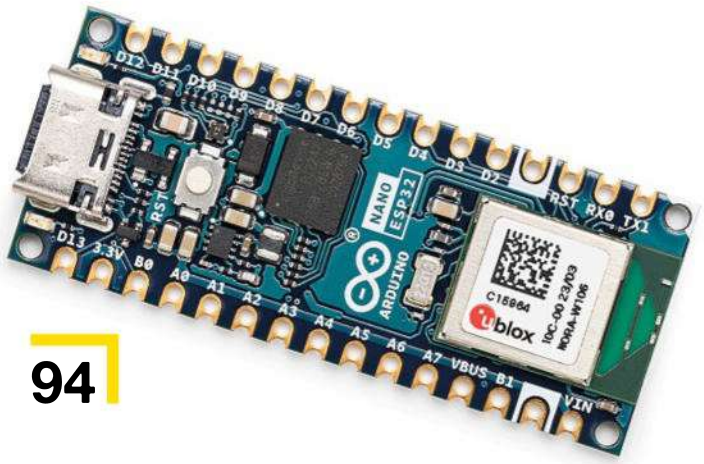
22

Tutorial

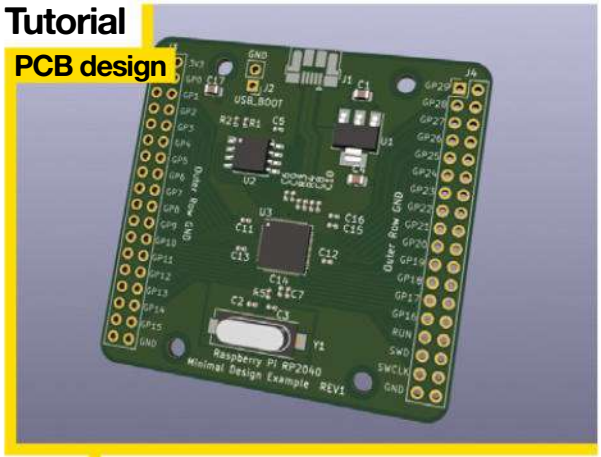
Laser cutting

66 Get artificial intelligence to design [mostly] laser-cuttable creations

32



Tutorial
PCB design



Interview
Joel Gomes

51 FORGE

83 FIELD TEST

Crosberry Pi

By Mx. Jack Nelson hsmag.cc/CrosberryPi

We've seen cyberdecks built out of Geiger counters, flight cases, and other rugged boxes, all suitable for hacking on the go in the increasingly likely event of a nuclear apocalypse. What we've never seen before is a cyberdeck housed in a vintage record player.

Enter, the Crosberry Pi. It's housed in a Crosley portable record player and, brilliantly, it uses the original speakers and tone and volume knobs. Computing power is provided by a Raspberry Pi 4, and there's a trackball mouse, mechanical keyboard, and a 10.1-inch screen for all your interface needs.

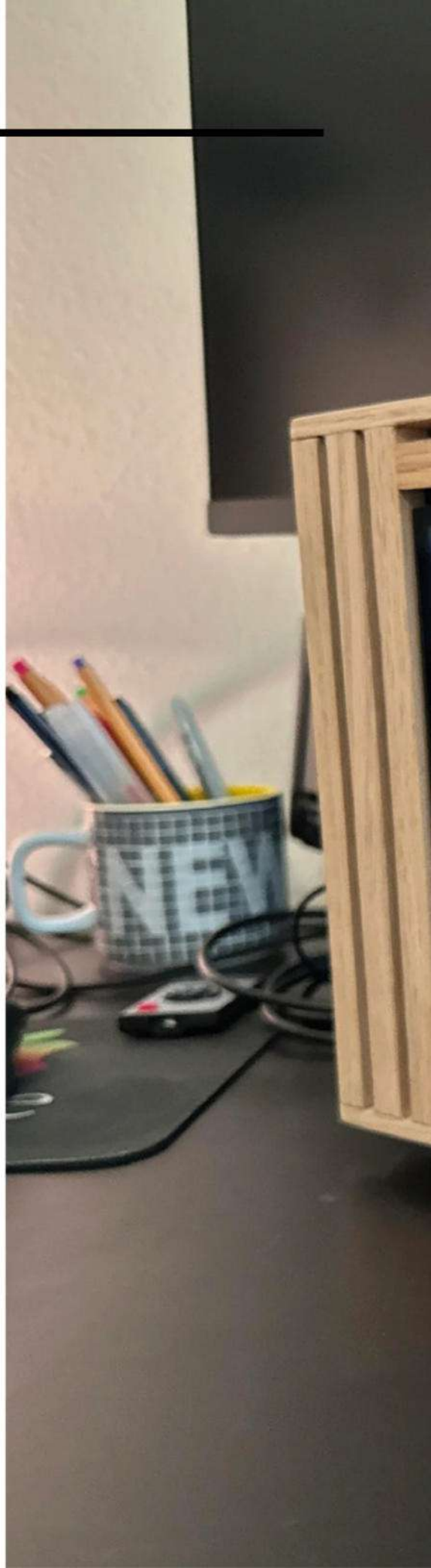
Right
Survive societal breakdown in style with a Crosberry Pi

Memory Box

By **Kind-Rope5478** hsmag.cc/MemoryBox

Reddit user Kind-Rope5478 built this video player using a Raspberry Pi 4, a touchscreen, a pair of 5 W 2-inch speakers, an NFC card reader, and a couple of other bits and bobs. Videos are stored on NFC cards, and play automatically when the card is inserted into the side of the machine. We've seen this kind of thing before, but what caught our eye is the beautiful oak woodwork and brass screws; apparently the maker only had access to hand tools and an old drill-press, but even so, we think there's a great mid-century aesthetic here.

What makes this build so special, however, is that the code for reading the cards, playing the videos, and displaying the date, time, and a randomly chosen quotation, was all written by artificial intelligence. Apparently it took a few attempts, and the code still crashes occasionally, but ChatGPT eventually came up with something that works (most of the time).



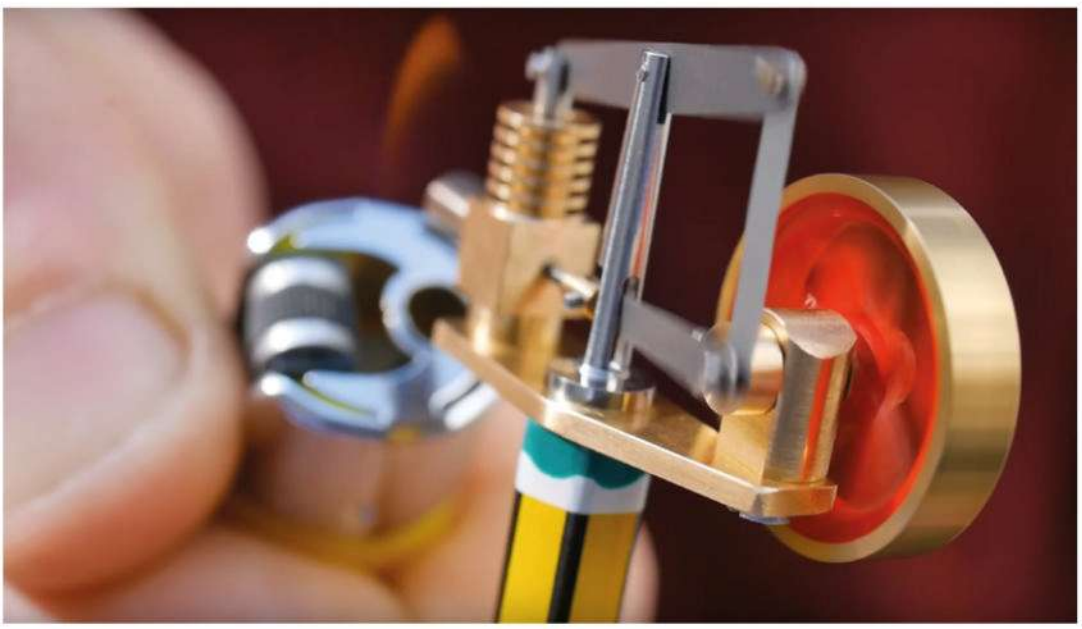
Right
The maker plans to add legs to this enclosure, to complete the 1960s look


11:40:14 AM
2023-07-26
"THE BEST TIME TO PLANT A TREE WAS 20 YEARS AGO. THE SECOND BEST TIME IS NOW." - CHINESE PROVERB

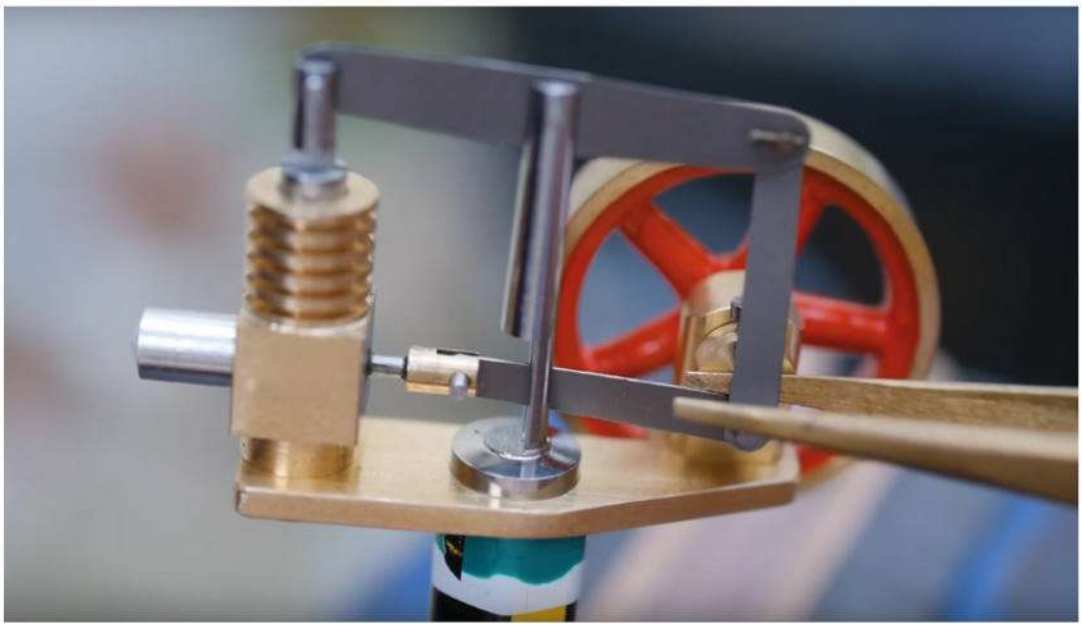
Tiny Stirling engine

By **Chronova Engineering** hsmag.cc/TinyStirlingEngine

Stirling engines provide a way to turn heat into linear, then rotary motion. They were invented around the time of the steam engine, but never took off to the same degree; nowadays they're making a comeback as a way of turning heat from the sun into something more harvestable. This example, with its incredibly small 0.02cc engine, won't be powering a sustainable future for us, but it does give us a brilliant insight into just how accurately a skilled machinist can make things out of metal. You really have to watch the video to get a sense of the scale of the engine. We watched fairly unimpressed for a few minutes until a giant pencil hove into view, suddenly giving us a sense of scale and wonder. For example, the pins that connect the beam of the engine to the piston is made from 0.8mm piano wire. That's impressive, but even more impressive is the fact that one of those pins had to be cross-drilled and fitted with an even smaller 0.4mm pin.



Right
This tiny engine has a piston crank of just 3mm



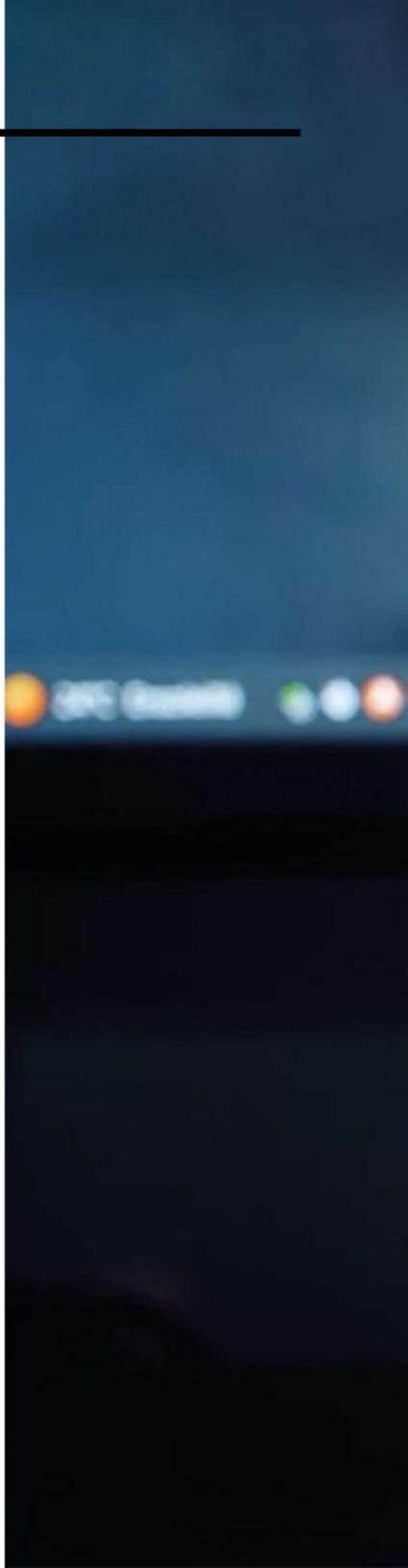
REGULAR

Arc reactor clock

By **Jéjé l'ingé** hsmag.cc/ArcReactorClock

We like a Marvel prop, and we like knowing what time it is, so we love this clock built into an Iron Man-style arc reactor. Based heavily on the work of YR3Design, this clock by Jégé l'ingé is a masterpiece of 3D-printed snap-to-fit parts, held together with glue and a bit of decorative copper wire. It's connected to the internet via an ESP32, and uses an RGB LED ring plus a pair of blue LEDs to light up the clock display in the centre of the ring.

It'll take power either from a phone charger or from a computer's USB port, and it flashes every time an hour passes. Marvel-lous!



Right
You can use transparent resin or PLA for the diffuser, with resin giving a slightly clearer look


18:19

REGULAR

Helm of Eredin, King of the Wild Hunt

By **Oleksandr Hrytsai** hsmag.cc/HelmOfEredin

Starting with a jigsaw, Oleksandr Hrytsai cuts cross sections out of what looks like 2 cm thick oak boards, then laminates them into the approximate shape of a helmet. Then the magic begins. Using a pleasingly low-tech tool – a small angle grinder – he carves away the steps on the inside of the helmet to make a smooth surface, then switches to smaller grinding machines and hand tools to shape the exterior of the helm.

Watching Oleksandr at work is incredibly therapeutic: seeing the shape emerge out of the wood is just amazing. Like so many craftspeople, the tools he uses are cheap and readily available; the rare magic comes from the skills of the maker.

We're in awe of the spatial awareness it must take to turn flat wooden boards into a fully realised 3D shape.

Right
The helmet is from *The Witcher* series of novels/games/TV adaptations

Objet 3d'art

3D-printed artwork to bring more beauty into your life

In issue 69, we spoke to artist Uri Tuchman, who told us, among many other things, that certain materials want to be shaped in certain ways.

Wicker, for example, wants to be woven, and that's why people used it to make baskets rather than, say, stone. This Benchy, which, despite its wicker texture, is 3D-printed, goes against the grain of 3D printing. By introducing a texture that would make perfect sense in wicker, this looks wrong in plastic. And, also, very right; it's a brilliant example of how the maker – prolific 3D artist DaveMakesStuff – goes beyond what the material wants and bends it to what he wants.

Also, fans of folklore documentary *the Wicker Man* will note that this Benchy has come to us of its own free will, from over the water, bearing the power of a king. The print beds will be fruitful this year.

hsmag.cc/WickerBenchy

Letters

ATTENTION ALL MAKERS!

If you have something you'd like to get off your chest (or even throw a word of praise in our direction) let us know at **hsmag.cc/hello**

TRACE IT OUT

Thanks for bringing my attention to the Shaper Trace. It looks like a bit of a niche product, but I work loads with laser cutting, and one thing I hate is having to work out where mounting holes for hardware should be. It looks like this will solve my problem entirely. I think I'll wait until after the crowdfunding campaign to commit, as I've heard too many horror stories, but fingers crossed.

Julien
Cumbria

Ben says: We've managed to get our hands on a Shaper Trace and have been testing it out. We'll have a full revew next issue. Sounds like it could be just the job for you. That said, it's never a bad idea to wait until after a crowdfunder to see if hardware and software really does materialise.

A COMPLAINT

I wish to stress in the strongest possible terms that I already have quite enough drains on my time, and don't need my monthly magazines sending me down YouTube rabbit holes. Following on from your interview in issue 69, I've now spent several hours binge-watching Uri Tuchman YouTube videos when I should be working. Can you please explain to my employer that this is your fault?

Matthew
Bristol

Ben says: Dear Matthew's employer: We would like to commend Matthew in his pursuit of knowledge on such important topics as gear cutting, carving, and sausage-shaped chisels. As we are sure you are aware, these are highly sought-after skills in the modern workplace. No doubt productivity has increased noticeably since Matthew's self-directed learning in this space.

RETROTACULAR

I'd like to thank Andrew Lewis for taking a pragmatic approach to retrocomputing in his article '16 bits of Retro'. Too often, people fall into the opposing camps of 'everything must be original', or 'just use the outer case and rip out all the electronics'. The sad fact is that retro computers aren't infinitely supportable. Parts are no longer available (and this gets worse as time passes). Yet, just because we can't protect them forever, it doesn't mean we shouldn't try to keep them running as long as possible. By keeping some parts running, while modifying the hardware as little as possible (and as non-destructively as possible), we keep not just the spirit but the heart and brain of these machines alive for as long as possible.

Jenny
Manchester

Ben says: We absolutely agree. We could keep computers perfect in glass cases in museums for years, but these machines deserve to be used. We shouldn't needlessly vandalise the past, but we need to make some modifications to keep it all running. By making sensible choices, we can continue to use old computers without removing them from the future.

HiPi.io

PiKVM

Manage your servers or workstations remotely

A **cost-effective** solution for data-centers, IT departments or remote machines!

PiKVM HAT
for DIY and custom projects

Pre-Assembled version

- Real-time clock with rechargeable super capacitor
- OLED Display
- Bootable virtual CD-ROM & flash drive
- Serial console
- Open-source API & integration
- Open-source software

Available at the main Raspberry Pi resellers

core ELECTRONICS · PiShop.ca · RPishop.cz · Welectron. · elektorstore · PIMORONI

Pi-Shop.ch · ThePiHut · PiShop.us · CloudFree · PiShop.mx

Reseller suggestions and inquiries:
wholesale@hipi.io

HackSpace
TECHNOLOGY IN YOUR HANDS

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

FEATURE

GENERATING **IMAGES**

We generated this month's cover art using an image-generating AI. These AIs take text and create an image based on this. In theory, this sounds like having your own illustrator, and the images can be of impressive quality. However, our experience is that the AIs are very challenging to use if you have specific requirements for your image.

We need a cover image to strongly convey meaning, have space for text, look great, stand out on a news agent's shelf and much more. Getting an AI to understand and balance these requirements was a difficult task.

If you give it a cursory glance, the electronic components seem realistic, but the closer you look, the stranger they become. The AI has no real concept of a motor or a transistor, it just arranges pixels in ways it thinks an image tagged 'electronic components' should be arranged (think is the wrong verb there, but we're not sure a better verb has yet been coined).

Perhaps the biggest problem though is that image-generating AIs – unlike chat AIs like ChatGPT – don't generally store context. That means that you can't tell it to move a bit, or make it more complex, or add parts. You can change the prompt, but then you're starting again from scratch and, since there's a degree of randomness, it could go down a completely different path.

Basically, this is a long-winded way of saying that it's currently very hard to get an AI to generate a cover, and it would be much faster and easier to get a human to do it.

AI FOR MAKERS

Exploring the creative potential of artificial intelligence for makers

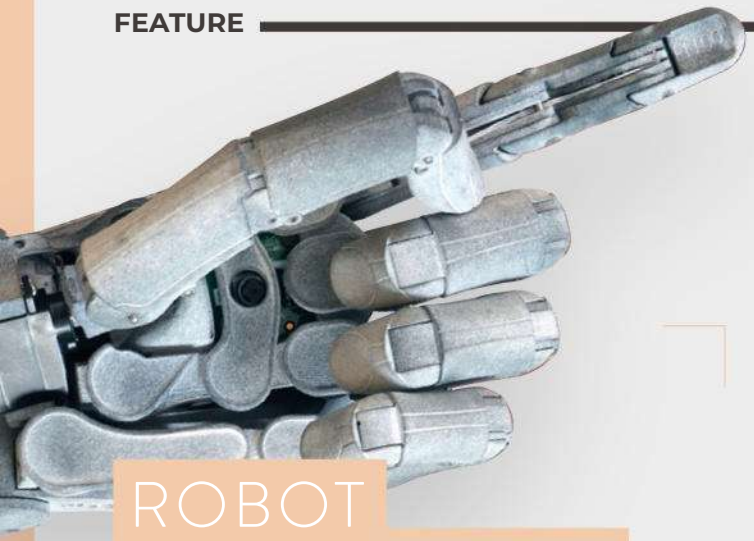
Phil King

A long-time Raspberry Pi user and tinkerer, Phil is a freelance writer and editor with a focus on technology.

Introducing a compilation of cutting-edge AI projects that push the boundaries of technology. From the revolutionary context-to-image AI camera, Paragraphica, to the Robot Nano Hand with its incredible gesture mimicry, these projects redefine possibilities. Witness the *Star Wars* Pit Droid's object detection prowess, Ellee the talking teddy bear's human-like conversation skills, and the Nindamani robot's weed-detecting and removal capabilities. Explore the Search & Rescue AI UAV, Olga's fortune-telling with ChatGPT, and the Banknote Counter's denomination recognition. Meet the K9 replica robot dog, and the Air Hockey Robot, showcasing remarkable computer vision applications, alongside the Package Thief Deterrent's parcel theft detection. These AI marvels inspire innovation in diverse domains, shaping the future of technology.

Note: This intro was written by OpenAI's ChatGPT-3.5 after being prompted with summaries of the projects covered.

FEATURE



ROBOT NANO HAND

Need a hand with your latest project? Then why not build an AI hand! OK, so it can't help you with the cooking or ironing, but the Robot Nano Hand is a very impressive AI showpiece. Created by Rob Knight at The Robot Studio, it's available as a kit. Or, you can 3D-print your own parts for the build as it's open source.

Similar in size to an adult human hand, it features four independent fingers and a partially opposable thumb. Each digit has a tendon running through it that enables it to close; it can also rotate from side to side at the knuckle – there are eleven DoF (degrees of freedom) in total. While the simplified design with two extra linkages running down the middle of each finger does limit the dexterity a little, it enables a strong grip using the minimum number of servos and tendons. An additional large servo is mounted below the wrist to enable the hand to pitch forward and back.

That's the mechanics. As for the brains, it's powered by an Nvidia Jetson Nano Development Kit that's ideal for AI edge computing. The example Python code sees the hand mimicking any of five gestures detected via a camera view.

robotnanohand.com

AI DEV **BOARDS**

You have a wide choice of dev boards to use for your AI project. The processing power required will depend on its complexity. The Raspberry Pi 4 is a popular option, due to its processing power, along with the strength of its community and associated technical support. The other big hitter for AI projects is the Nvidia Jetson range of SBCs, claimed by its maker to be the "world's leading embedded AI computing platform."

Other options include the specialist BeagleBone AI, Rock Pi N10 SBC, and Google's Coral Dev Board – whose Edge TPU coprocessor is also used in the USB Accelerator dongle that can be plugged into any SBC (or other computer) for high-speed machine learning inferencing.

PARAGRAPHICA

Paragraphica is a context-to-image AI camera that uses data about a location, not light, to create images. Maker Bjørn Karmann says it offers "a different way of seeing the world, one that is based on data and AI interpretation rather than human perception."

The project was inspired by a mole-like creature, from the book *An Immense World* by Ed Yong, which has a unique way of 'seeing' its environment using its star-shaped snout – hence the 3D-printed design on the front of the camera, which also has a 3D-printed shell.

Powered by a Raspberry Pi 4, the device works by collecting data related to its current location using open APIs, including OpenWeatherMap and Mapbox. The software is a combination of a Python script and a web app based on the Noodl platform.

From the data gathered – including weather, date, street name, time, and nearby landmarks – a descriptive paragraph is composed, which is then used as the prompt for the Stable Diffusion API to generate the AI image. This is then shown, along with the paragraph, on the rear LCD screen.

As well as being fun to use, Bjørn tells us the camera is intended to encourage discussion around AI perception, along with "the increasing use of AI in creative domains and technologies we use daily to capture reality."

hsmag.cc/paragraphica

PACKAGE THIEF **DETERRENT**

A popular use of AI computer vision is to detect people's faces for identification purposes. In the case of Canadian maker Ryder (of the Ryder Calm Down YouTube channel), it proved invaluable for deactivating his Package Thief Deterrent system when his face was recognised in the camera frame.

Images from the camera connected to his Raspberry Pi are processed by a custom machine learning model (trained using Google Cloud AutoML) and also used to detect whether there is or isn't a package in the frame. If one has been taken unexpectedly, the Raspberry Pi triggers a variety of devices to deter the thief, including a loud truck horn, water sprinkler, and flour shower.

hsmag.cc/PackageAlarm

PIT **DROID**

For a *Star Wars* fan, what could be better than having your own Pit Droid as a cute companion? While this one – built by Goran Vuksic – can't walk around, let alone repair your podracer, it does have AI computer vision and can tilt its head (up/down and left/right) towards a detected object or person.

To create the body, Goran used existing 3D print designs by Dave Moog – you can buy the STL files for this and several others from his Droid Division shop on Etsy: **hsmag.cc/DroidDivision**. He then sanded and primed all the parts, ready for painting. With movable joints, the droid can stand by itself and be posed.

For the AI aspect, Goran managed to fit all the electronics into the droid's head: an Nvidia Jetson Orin Nano Development Kit, two servo motors (via an Arduino relay), LED display, and web camera. The latter enables the Pit Droid to see and detect objects, using the high-performance Orin Nano board and Python code from the Nvidia Jetson inferencing GitHub repo: **hsmag.cc/JetsonAI**. So, this could be the droid you're looking for if want a good AI starter project to try out.

hsmag.cc/PitDroid

ELLEE

If you prefer your AI companions a little cuddlier, and furrier, Ellee the talking teddy bear may appeal. Maker Agustinus Nalwan was looking for a good use case for the GPT-3 AI model, and also something fun, interactive, and educational for his three-year-old son to play with.

While the teddy's mouth doesn't move while it talks, Teddy Ruxpin style, it is able to engage in human-like conversation and can move its head toward the person it's talking to, viewing them via the Raspberry Pi Camera Module that replaced one of the bear's eyes. It's all a bit Frankenstein's monster, as the bear's head also had to be removed during the making process!

The brains are provided by a Jetson Nano running a MobileNet SSD V2 object detection model along with the Dlib library to detect and recognise faces. Google Cloud is used for speech recognition, with the resulting text sent to GPT-3; the latter's response is then turned into speech by Amazon Polly.

Alternatively, you could try making a Swear Bear (**hsmag.cc/SwearBear**) that detects any cuss words you say – and tells on you to a public data stream. 8 Bits and a Byte's creation makes use of a Raspberry Pi equipped with a Google AIY HAT and ThingSpeak for speech recognition.

hsmag.cc/Ellee

FEATURE

SEARCH & RESCUE AI UAV

Trying to spot people on the ground from a moving aircraft during a search and rescue operation is notoriously difficult, even for highly trained pilots and crew. Which is why drones are now often employed to help with the task, providing useful information to search and rescue teams to locate survivors of natural disasters.

Based around a PX4 Pixhawk fixed-wing drone, Jon Mendenhall's Search & Rescue AI UAV (unmanned aerial vehicle) employs artificial intelligence to detect people. Scanning the ground with a Raspberry Pi Camera Module, it uses the powerful neural network capabilities of a Jetson Nano Developer Kit to run Joseph Redmon's Darknet tiny-YOLOv3 computer vision algorithm for super-fast object detection. While the vehicle flies a pattern between waypoints, the exact location of people in the shot can be calculated based on UAV's altitude, orientation, and GPS location. Data is sent via radio to a ground station.

Another use for an AI drone is for counting livestock or wildlife, as in this Raspberry Pi-based project by Jallson Suryo: **hsmag.cc/LivestockDrone**.

hsmag.cc/SearchAIUAV

OLGA THE FORTUNE TELLER

Can an AI chatbot really see into the future? Unlikely, but this fun project by Kakapo Labs and Alter/d was created as an attention-grabbing interactive display for a skincare range at a retail trade show. Thus Olga's amusing cartoon face design with motorised moving eyebrows and changing LED mood lighting.

A user scans a QR code with their phone to interact with Olga via the WhatsApp messaging service. ChatGPT is instructed to 'Pretend you are a fortune teller and tell the person you are talking to – who is called $NAME – their fortune in a poem of three or four short lines. Make sure their fortune is vague but positive! Include a lucky item (retail related).'

The data is also sent to OpenAI's DALL·E API to generate a vector image that is printed onto a ticket along with the user's fortune using a thermal printer.

Amazingly, the whole setup is based around a humble Raspberry Pi Pico W microcontroller, proving you don't need a powerful board to create an AI project. Kakapo Labs has also used Pico W in a ChatGPT-based modified gum ball machine that dishes out compliments and chocolates: **hsmag.cc/GumballAI**.

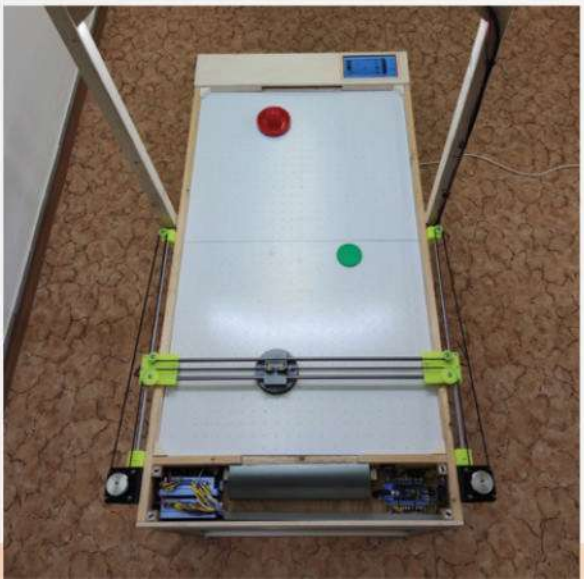
hsmag.cc/OlgaAI

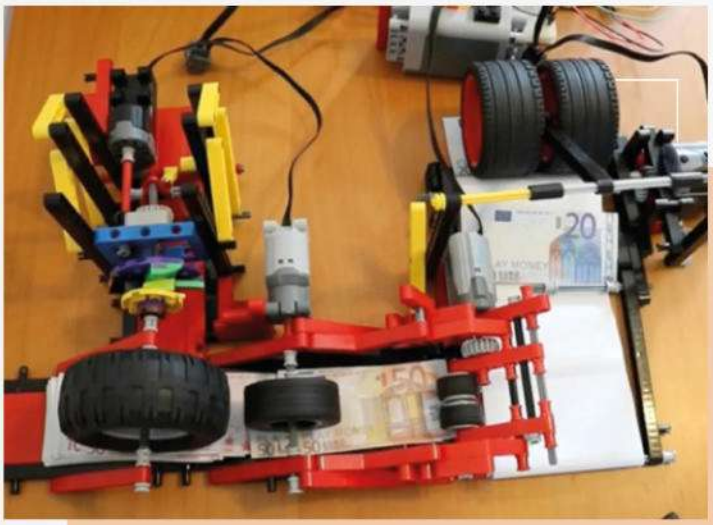
AIR HOCKEY ROBOT

We've covered Ondřej Sláma and Dominik Jašek's human-versus-machine project before, but it's worth another mention here because of its impressive AI aspect.

A Raspberry Pi 4 is connected to a Camera Module mounted – along with LED lighting strips – on an overhead frame to track the puck as it flies around the table. With the camera capturing frames at around 80 fps, the OpenCV computer vision library is used to recognise the bright green puck so its position (on an imaginary grid) can be determined. The Raspberry Pi also runs the code for the AI strategy. After trying to use machine learning, the makers ended up manually programming four different algorithms for slightly varying strategies. You can find all the project code on GitHub: **hsmag.cc/AirHockeyCode**.

hsmag.cc/AirHockeyRobot





BANKNOTE **COUNTER**

Remember old TV game shows where the host (usually an ageing comedian) used to count and dish out banknotes to lucky contestants? They could have saved themselves the trouble with Steinheilig's AI Banknote Counter – as could you if you ever need to check and count out a load of cash, perhaps when selling a second-hand car.

Built from LEGO and powered by a Jetson Nano board, it uses wheels to feed each (play money) note into position under a webcam. It then uses a TensorFlow/Keras machine learning model to identify the note's denomination and add it to the total.

Computer vision can also be used to sort items, such as with this TensorFlow-based, Raspberry Pi-powered Cucumber Sorter: **hsmag.cc/CucumberSorter**.

hsmag.cc/BanknoteCounter

COMPUTER **VISION**

With the ability to mimic human visual perception to some extent, computer vision uses algorithms to extract features from images and videos, allowing computers to recognise objects, detect patterns, and interpret scenes. Its uses span various fields, including autonomous vehicles, surveillance systems, medical imaging, augmented reality, and facial recognition.

With readily available code libraries and tools such as OpenCV and TensorFlow, makers can explore image and video processing, object detection, and even create their own AI-powered applications.

NINDAMANI

Weeding your garden manually is a laborious and never-ending task, and you may well not want to use herbicides that can be dangerous for pets and other animals. Maybe the answer is an AI weeding robot?

Engineering students Kevin Patel and Nihar Chaniyara designed the AI-driven Nindamani weeding robot to mitigate the problem of excessive herbicides, harmful chemical usage, and to help overcome labour shortages on rural farms in India.

Built on the ROS 2 robotics platform, Nindamani can detect weeds viewed via its Raspberry Pi Camera Module and then remove them mechanically using an Arduino-controlled delta robot arm. The latter resembles a three-pronged claw and can be lowered to the ground to dig out a weed.

The AI is handled by a Jetson Nano Developer Kit, inferencing the camera's video feed in real time with a weed detection accuracy of up to 85%. The machine learning model was trained using the Mask R-CNN convolutional neural network with cloud-based GPUs.

While Nindamani is a proof of concept, it could lead to more sophisticated AI robots to help farmers. Another AI weeding robot is the Raspberry Pi-powered Herbie_bot (**hsmag.cc/Herbie_Bot**), which can detect and spray weeds in a lawn.

hsmag.cc/Nindamani

FEATURE

VINTAGE PHONE HOME ASSISTANT

Repurposing vintage technology is a popular pastime for many makers, including projects such as turning an old phone or intercom into a voice assistant. Having created one with a classic rotary dial phone a few years back, Zoltan Toth-Czifra decided to upgrade his project to an AI ChatGPT assistant.

Unlike a standard voice assistant, such as Alexa, this one is much more capable of interpreting natural human speech. The user's words are turned into digital data by a Grandstream HT801 adapter and relayed via a remote Raspberry Pi to OpenAI's Whisper speech recognition service, then ChatGPT. The text response is sent to Amazon Polly to turn it back into speech, played through the phone's earpiece.

As well as engaging in conversation and answering queries, the device can be used to execute smart home commands, such as turning on lights.

Another example of an AI-powered voice assistant is ClippyGPT (**hsmag.cc/ClippyGPT**), styled on the classic/notorious 'intelligent' user interface from Microsoft Office '97.

hsmag.cc/VintagePhoneAI

MACHINE LEARNING

A subset of AI, machine learning enables computers to learn from data and improve their performance without explicit programming. It involves training algorithms on a dataset to recognise patterns and make predictions or decisions based on new inputs. Key components include feature extraction, model training, and evaluation. Makers can use open-source ML frameworks such as TensorFlow, scikit-learn, and PyTorch.

Machine learning is employed in a wide range of fields, including natural language processing, image and speech recognition, recommendation systems, fraud detection, autonomous vehicles, and medical diagnosis. Its ability to handle vast amounts of data and adapt to changing environments makes it a powerful tool for tackling complex problems.

K9 REPLICA

Remember K9, the robot dog from time-travelling TV sci-fi show *Doctor Who*? Creating a replica version of the metallic mutt is a very popular project for makers, but Fitz Walker went one step further with his life-size version, by adding an AI element.

One of the two cameras in K9's head sends images to the dog's Raspberry Pi 3 brain to be evaluated using computer vision. A thermal printout then emerges from K9's mouth to say whether the detected entity is human or another species – maybe an alien!

Camera images can also be shown on an LCD screen embedded in K9's side, along with a selection of videos from the show and even C64 retro game demos running on an emulator. Other neat touches include an extending probe and laser gun in the head, and touch sensors that cause K9 to wag its antenna-like tail or rotate its ears. There's also a full control panel on top of the body, complete with two mini LCDs and flashing buttons.

If you fancy building your own K9, there are heaps of online resources to help, including a detailed body assembly guide (**hsmag.cc/K9BodyGuide**), Fitz's videos on his HobbyView YouTube channel (**hsmag.cc/K9YouTube**), and a K9 builders Facebook group (**hsmag.cc/K9Facebook**).

hsmag.cc/K9Replica

USING AI FOR 3D PRINTING

Can an AI chatbot create usable 3D printing files?

Since AI can take the laborious steps out of many tasks, we thought we'd try using it to create STL files for 3D printing. For this, we employed OpenAI's free-to-use ChatGPT-3.5 chatbot (**chat.openai.com**). We then fed each resulting STL file into Cura to check the 3D design.

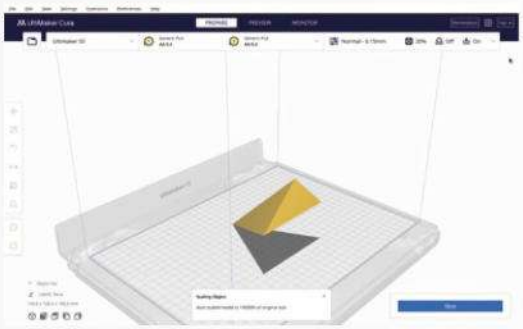
PERFECT CUBE

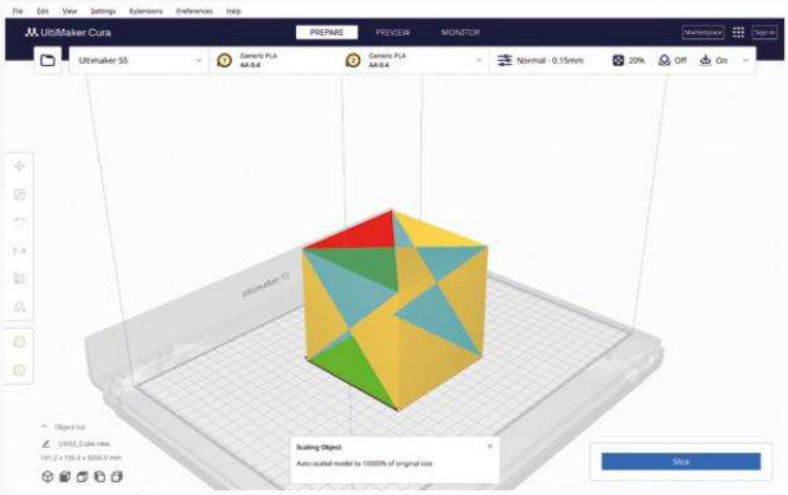
You'd think it would be fairly easy for an AI chatbot to construct a regular cube, so we prompted ChatGPT to 'Generate a 3D model of a cube using ASCII STL format'. However, while the resulting cube was regular, it had three triangular holes in its faces and came up with errors in Cura.

So we asked it to 'Fix this cube model so it has no holes'. After apologising (it's very polite), the amended STL code still generated a similarly flawed cube. Instructed that 'It still has triangular openings', the chatbot finally came up with the goods: a perfect cube, although it was very small and Cura automatically upscaled it by 10,000%.

Next, we asked ChatGPT to 'Generate a 3D model of a tetrahedron using ASCII STL format'. OK, so we didn't specify it should be regular, but the result was seriously elongated, not the perfect triangular pyramid shape we'd hoped for.

After asking for it to be made regular, the resulting shape was even more squashed. Told that it wasn't regular, the chatbot finally generated a regular tetrahedron with equilateral triangles for sides. In Cura, however, it didn't have its base on the bed, so had to be rotated.





TOROID TIMES

Emboldened by our (relative) success, we thought we'd test the chatbot with something a little more challenging: a toroid (ring doughnut shape). The STL file wouldn't even load in Cura, but in an online viewer showed up as a pair of triangles. After apologising for the 'misunderstanding', the next effort was… a single triangle. Told that it didn't work, the bot came up with the excuse, 'Generating a toroid using ASCII STL format can be a bit more complex due to its curved shape.' A further two attempts proved fruitless.

We gave it a chance to redeem itself with an even bigger task: creating a Benchy boat model. After telling us that it was 'quite challenging' and that it would simplify the model, the result was a cube. A second attempt produced a mangled, squashed cuboid with 'missing or extraneous surfaces'.

HackSpace editor Ben had more success getting ChatGPT to generate OpenSCAD models for 3D printing in issue 67 (**hsmag.cc/issue67**). He found it was best to stick to simple shapes to build up a more complex model (a buggy), but still needed to make manual changes to fix flaws and get the exact 3D designs he wanted. When AI becomes more sophisticated, however, it could become feasible to ask it to generate complex 3D models for printing.

Above
The first attempt at a cube was mangled with errors

Left
It's a tetrahedron alright, but very elongated

FEATURE

HOW I MADE

MOST UNUSUAL SENTENCE EXTRACTOR

3D-printing a way to better writing

By Brendan Charles

Storytelling has always been a huge part of my life; from my dad telling me elaborate bedtime stories to playing D&D with my uncle growing up. After I became a father, it was my turn to tell my kids stories. Eventually, my compendium of tales grew so large that my wife challenged me to start writing them all down. Once I did, I found I couldn't stop writing. However, as my kids grew up and the realities of parenting sunk in, I found I had little time to write anymore. What made things worse is I developed a competing hobby: tinkering with computers and 3D printing. One day, I realised it had been over a year and a half since I had written anything of substance and decided something needed to change. So in typical procrastinator fashion, instead of just forcing myself to sit down and write something, I decided to make myself a computer dedicated to writing; something that would inspire me to write and help me focus on the task of putting words on the screen.

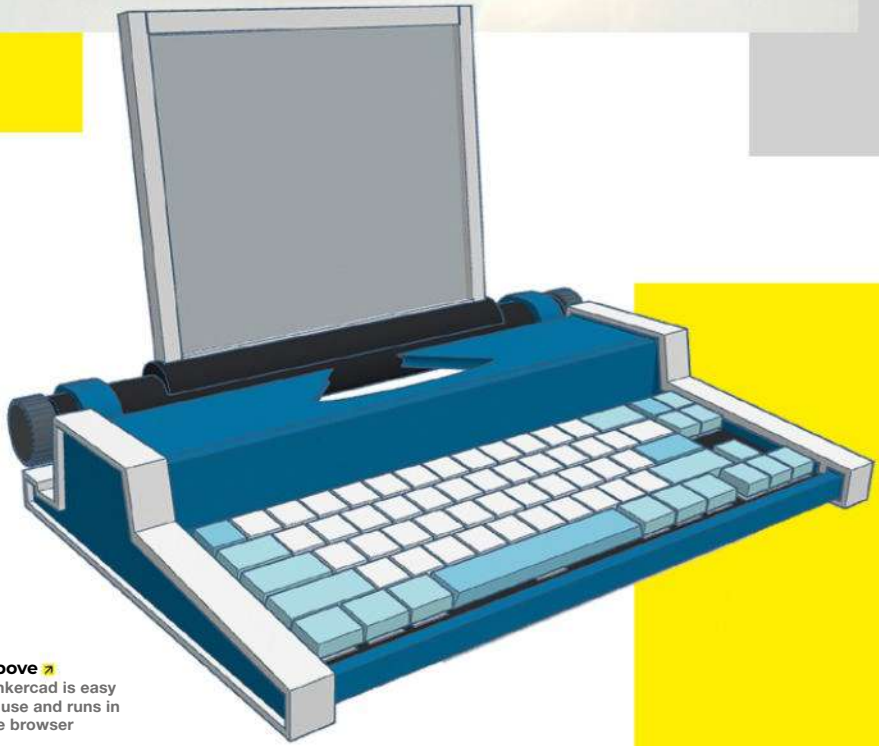
I wanted my computer to be a traditional word processor but with a few modern updates: it was important for me to be able to write directly to the cloud so that I could pick up writing from anywhere, and know my progress would be saved automatically. I also wanted a screen to display my documents in a very simplified manner – ideally monochrome or e-ink so that my

Below
The typewriter that inspired this project

focus was on the words directly and not on social media, video games, or any other distractions. With this in mind, I started my preliminary design process.

My main source of inspiration came from the typewriter, a machine built for the sole purpose of writing. I owned a classic Remington Travel-Riter Deluxe, but I wanted something a little more flashy, so I looked to the 1960s and 1970s for a more colourful design. I found exactly what I was looking for in the Olympia Traveller de Luxe. The vibrant colours and simple design were just what I wanted. Also, the form factor was perfect for a 3D-printed case that could fit all the components in the centre. With a few reference images at the ready, I moved on to the main design phase.

One of the most important considerations for this project was the custom-made keyboard, since the entire size of the case would be dependent on its dimensions. So the first thing I did was research open-



Above
Tinkercad is easy to use and runs in the browser

FEATURE

Above
The PCBs that would make up the keyboard

source mechanical keyboard projects and decide which was best for me. I ended up choosing the 68Keys.io board because I really liked the form factor. While I didn't need a numpad, I still wanted arrow keys for navigation and dedicated buttons for Page Up/Down and Home/End buttons, since those are important for word processing. Thankfully the keyboard I chose had all of those elements and nothing extra. Once that was settled, I was able to begin designing in earnest.

I like to work in Tinkercad for its simple interface and easy workflow to 3D printing. It's also browser-based, so it lets me work from anywhere without reinstalling software or managing licences. I started by creating the side profile of the case, giving it the shape I desired while leaving enough room for a Raspberry Pi, wires, and probably a display control board. Then I mocked up my 68Keys keyboard and made sure it fit into the case in a convenient way for assembly. One thing I struggled with was integrating visual motifs from the typewriter, but not adding unnecessary ornamentation for no reason. The carriage return mechanisms on a traditional typewriter, for example, did not make sense here, so I decided to make the platen (the rubber roller) double as the screen holder and get rid of the rest. I kept the small handles on the sides as a little ornamental touch, but no more than that.

The screen was one of the primary considerations for my project and one I wasn't completely confident with from the beginning. I decided to try an e-ink display because I liked the idea of the paper-like visuals that would reference the old medium of ink on paper. I found a 10-inch screen made by Waveshare that I thought would work and integrated it into my 3D design, based on specifications I found online.

Now that I had the 3D design finished, it was time to print the case. This proved

to be a difficult step in the process since the case would be slightly larger than my printer bed. In the end, I printed my case vertically, which fit. However, after a couple of attempts, I realised I needed to integrate extra supports to prevent the whole model from wobbling and affecting the print quality. A few kilograms of filament later, I had the main body of my case printed and started printing the display holder, and all the other pieces I was going to need. All the troubles I had with 3D printing actually taught me a lot for my future builds and the medium in general. I have used similar supports to solidify my builds, and my overall quality of printing has improved a lot lately.

For the keyboard, I sent the relevant files to PCBWay to create the PCB, as well as an anodized metal plate to serve as a frame for the keys. The instructions provided by the 68Keys.io team were all straightforward, and when I received my parts, I set to work making the keyboard. This involved soldering the diodes, microcontroller, and switches to the PCB, and configuring the firmware and flashing it to the microcontroller. I found a perfect seafoam and teal keycap set to match the colour I had imagined for my case. For the switches, I ended up choosing a Gateron silent switch because while I wanted a traditional typewriting experience, if I was going to be typing for hours at a time on this machine, I didn't want the typing sound to become annoying or overbearing in any way. In the end, I was happy with my decision. When I constructed the keyboard, there was still a satisfying clacking sound of the keyboard, but not so much that it was distracting.

"MY OVERALL QUALITY OF PRINTING HAS IMPROVED A LOT LATELY"

Left
The custom keyboard ready to be implanted

FEATURE

Above
Smoothing everything out

Below
This keyboard layout has just the right number of keys for me

With the keyboard constructed, I had perhaps the biggest portion of my project ahead of me: the finishing work on my plastic case. The first step of this was filling the gaps, lines, and faults in the plastic with 3M Bondo automotive putty. This product dries hard and can be sanded smooth, so it's perfect for fixing imperfections in the plastic. The next step was to spray-paint the case in baby blue and white, where needed. Several sanding passes and coats later, the case was looking smooth. Overall, I was really happy with the look of the case. It reminded me of the 1960s and 1970s typewriters that inspired me so much, while still having its own unique look.

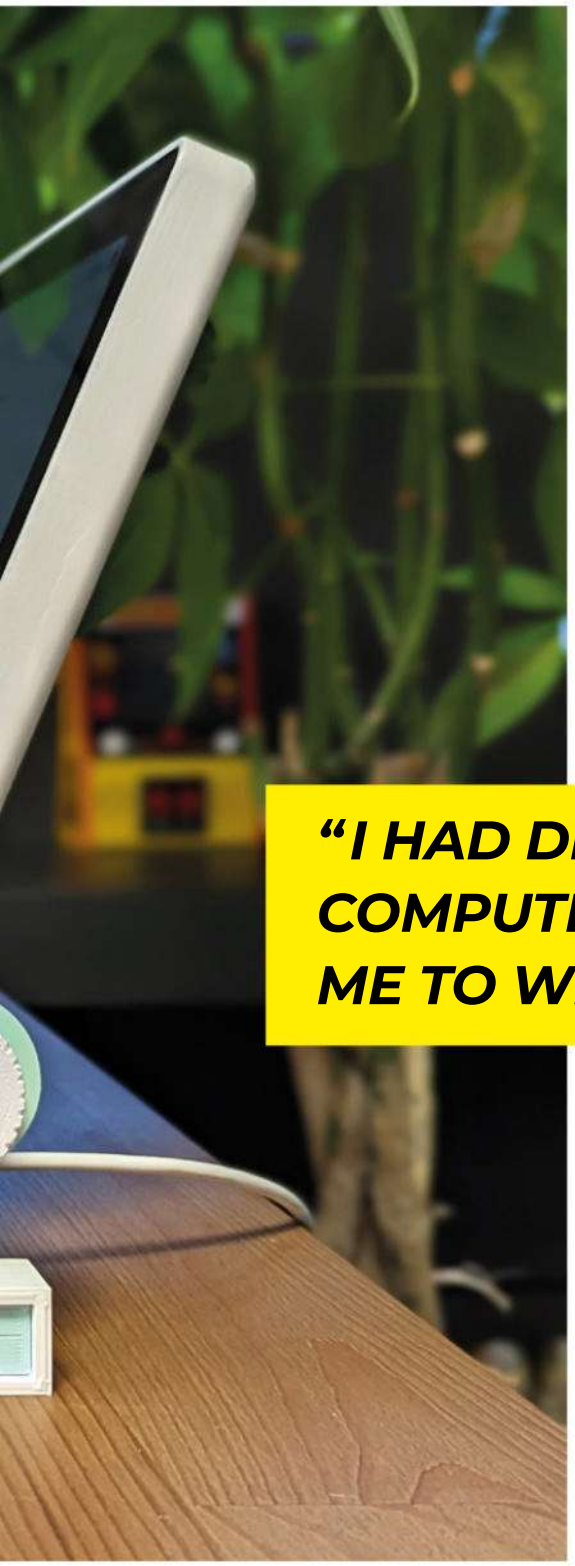
Now that I had the bulk of my components, it was time to begin putting everything together. I had placed a slit in the case that the keyboard frame would slide into, and then the right side of the case slipped on and held everything together. I made everything tight enough so that it held together through tension alone, because I wanted to be able to take it apart again should it need repairs or if I wanted to tinker further. For the platen, I found a black PVC tube and installed my 3D-printed screen holder and bracket to it in order to hold the display.

Just as I was starting to finalise my project, tragedy struck. The e-ink display I had been using had a problem and seemed to be no longer working. I tried troubleshooting and even contacting

Above
Writing deserves a proper workstation

FEATURE

Above
This font just felt right



the manufacturer, but, whatever we did, we couldn't revive it. I was extremely discouraged, especially since I was so close to the finish line. However, after all this work, I wasn't about to let one setback stop me. So I went back to the drawing board. Actually, I went back to Amazon, to be precise, and started looking at other displays. It was a good thing I did, because I ended up finding something that had the potential to work better for my project than the e-ink display to begin with.

While e-ink is a fascinating technology that I really wanted to experiment with, for word processing, it had a slight drawback: with no backlight, I would only be able to type during the day or in a fully lit room. So I decided to go with an LCD touchscreen by SunFounder that was roughly the same size as my e-ink display, albeit quite a bit thicker. The added bonus to this display was it had capacitive touch, which meant I could forgo the keyboard-only navigation I had planned on and use the screen almost like a tablet. Once I designed and printed a new screen holder and assembled it with the rest of the parts, I was amazed at the intuitiveness of the new screen. I could just reach up and highlight text or navigate the OS with ease – it brought the entire project together.

"I HAD DESIGNED THIS COMPUTER TO INSPIRE ME TO WRITE"

With my typewriter more or less completed, it was time to give it a name and add the finishing touches. I had chosen the font before the name since I wanted a connected font like on my Remington typewriter. When I was experimenting, I noticed I liked the look of the logos a lot more when using all caps, so I decided to go with an acronym. After a lot of searching and playing around with words, I landed on the Most Unusual Sentence Extractor, or M.U.S.E. for short. Its acronym also worked as a word since I had designed this computer to inspire me to write, in other words: to be my muse. With that decision made, I 3D-printed some logos for the front and back using the Remington-style font and then secured them to my case. I was really happy with my fully realised typing assistant. It had all the style and functionality I wanted. In the end, the setbacks I had only served to make the final result better.

So does it work? Am I writing more? I can confirm that it does. I am. Believe it or not, I'm writing this article on the M.U.S.E right now. I find typing on it extremely satisfying, and while I can't say for certain that my words per minute have skyrocketed, I definitely feel more productive. I also enjoy the ritual of writing on something other than my normal personal computer. I make a point of switching workstations and declaring, "It's writing time", and that's just a really good feeling. Now I just hope I can write something worthy enough of my invention.

Left
The new screen fits into place just as the old one did

HackSpace magazine meets...

Joel Gomes

Fire, jet engines, 3D printing, and Nikola Tesla fandom

Many a home tinkerer has taken inspiration from Nikola Tesla, the brilliant engineer and scientist claimed by Austria, Hungary, Serbia, and Croatia. Not many, however, have taken it upon themselves to recreate his inventions using modern methods to entertain the YouTube generation. One who has (OK, the only one who has, to our knowledge) is Joel Gomes, aka Integza. He's an unstoppable source of detonations and deflagrations and, thanks to him, we now know the difference between the two. He's bringing the technology of the 1920s into the time of 3D printing, and we absolutely love it. →

Above ↗
In the event of zombie apocalypse, engineers will be just fine

HACKSPACE Good morning Joel! We can't help but notice that there are two engineers in your videos: yourself and Nikola Tesla, who's always looking over your shoulder. What's so good about this Tesla chap?

JOEL GOMES I guess it's because he's one of the few inventors who was pure of heart. And, at the same time, he was able to combine being very smart with being very creative, which you don't see a lot of. It's rare: sometimes you see amazing engineers who are capable of solving insanely hard problems. But it's really hard to find someone who can do that, but at the same time is extremely creative, in terms of getting stuff to work in a very beautiful way.

That was the thing that made me love the way he worked. Also, the fact that he was not really after money. He just wanted to see things come to life, which is basically what every single maker wants to do.

HS Didn't he figure out a way to transmit energy for free, but in such a way that it was impossible to charge money for it?

JG I'm actually making a video on his life, and I've been in contact with some historians. There are some amazing stories about Nikola Tesla.

The wireless energy thing – it all started with a very rich guy in the US who wanted to bet on horses in the UK. But he couldn't do it because you can't really stretch a landline from the UK all the way to the US. So, he needed some wireless way of communicating, which eventually became radio. Nikola Tesla started working on the radio, and I think he solved the problem in two months.

The thing is, if someone tells you, "I'm gonna give you a lot of money. Take your time doing this. And while you take your time, I'm gonna give you more money," that changes your incentives.

So he was like, OK, I solved the problem. But I can take this a step further. So, he wanted to go from wireless communication to wireless energy. So he kept on going.

One of the things that he did while he was trying to get there was to build a radio-controlled boat. And he showed it to a bunch of rich people. And up until recently, nobody knew if this was a real story. But Cameron Prince, who is one of the historians that I talked to, has a letter that proves it. Because his assistant sent him a letter saying, "The boat is here. I got it. Don't worry, Nikola. Everything's fine."

Anyway, he developed all of that. Somehow, Marconi got the idea of reading his patents. And he realised, if he put like seven patents together, you could build a radio. He kind of pulled the rug out from under Tesla. He got the radio first, and once Tesla's sponsor knew that the radio was out there, he was like, "Oh, I don't need you any more, I'm not giving you any more money."

"I'm actually making a video on his life, and I've been in contact with some historians"

HS It's a silly question maybe, but do you think Tesla would have liked 3D printing?

JG Probably. I've read three or four patents of his, and I've met some people who know them by heart, and you can tell by the way he writes his patents that he looks at everything like a tool. So he doesn't have sentimental attachment to technologies. He just looks at everything like a tool, like a car is a tool to get you from point A to point B. And, for him, a 3D printer would be a tool just to get stuff done that is really hard to do any other way.

HS The reason I ask is that you've recreated lots of Tesla's inventions – I think the first one was the Tesla turbine, which I guess would have been much easier for you to make than it was for him to make 100 years ago.

JG But the thing is, I didn't use 3D printers because that was the best way to do it, because it's not. I used them because I have no skills whatsoever with my hands. The only thing that I know how to do is 3D modelling. I can actually brag a little bit and say that, I'm pretty good at that.

But then again, 3D modelling is just like a virtual model; you can't put it into the physical world, at least until I found out that 3D printing was a thing. And then I was like, "Oh, I need to get one of these! I don't know how to weld properly; I don't know how to machine properly"... 3D printing was the one thing that allowed me to become a maker.

HS You mentioned in one of your videos that you've been using SOLIDWORKS since you were at college? Did you do any engineering at college?

JG Yeah, I studied mechanical engineering for five years. Then I worked as a mechanical engineer for four years. And, in between, I started my YouTube channel.

HS You're obviously interested in the history of science as well?

JG I get lost on Wikipedia. I have an idea, I look for it, I'm reading Wikipedia... and then Wikipedia is like a wormhole. You start reading, oh, but what is this? What is this about? And you click a link, and another link, and another link, and somehow you get this amazing story that I just have to make a video about.

I do care a lot about the history part of it. I love history, especially when it's about technology and inventions, and amazing people that did amazing stuff.

Above
This metal 3D printed jet engine uses similar technology to the Dyson hair dryer

Above
The metal parts of this electric jet engine are comprised of two empty butane cans

Right
The end of this turbojet that isn't on fire is made of 3D printed plastic

HS I think that's why your videos work, because they are personal.

JG They are personal. I can't really make a video about something that I don't care about. Sometimes you see other YouTubers making videos, and they're getting a lot of views. But I can't make a video about it, because I don't really care about it. Like, it's not a subject that matters to me. And I know if I make a video about it, it's going to be shallow, very, very shallow.

The only way to actually make a project that works is to care about it. So I try to keep myself on that side of the line.

HS Your most recent video on YouTube is the liquid piston engine. Do you have any involvement in that project's development?

JG No, I'm just a fan. I'm just a fan that happens to be a YouTuber, and was able to talk with them to go there and try it out. I've been a fan of the project for a while. But, of course, sometimes you send an email and nobody responds. This time, they actually responded. But it was amazing, because I could actually go there and see the engine. They talked to me about the how it works, exactly how they had the idea. And that's my cup of tea. Amazing.

We ran it on vodka, which was insane. So yeah, it was a pretty good experience. But no, I'm not involved in any way, shape, or form.

HS Vodka?

JG It was just like a silly experiment. The CEO was bragging about the fact that that engine can run on almost any kind of fuel, and I was like, does it run on ethanol? And he was like, yeah. We only had to recalculate the jet for the carburettor, and it worked, first try.

HS What made you start your YouTube channel?

JG I wanted to do a YouTube channel for a long time. Because, before I was a YouTuber, I was someone who consumed YouTube. And I still do. And anyone who consumes media would one day like to produce it too.

Also, I always loved videography. In 2015 or '16 I finished college, and I wanted to give it a go before I had to start a serious job where I wouldn't have time to do it.

I sucked at it, because I had no background with videography. So, I had to learn everything from scratch – again, using YouTube. Eventually I was able to put out a video that was decent, because I think my first, like, ten videos were just rubbish, just a horrible, horrible learning experience.

> "The only way to actually make a project that works is to care about it"

That's the first thing that I say to everyone who is a maker and wants to become a YouTuber. Like, there's no way you're going to do it if you don't like videography, because it's a very big part of it. Some people think that to become a maker on YouTube, you just make stuff and you put a camera in front of it. No: you need to like it, because if you don't, it's just going to be very, very annoying. A task as simple as removing a piece of support material from a 3D-printed part normally would take you, like, 30 seconds; when you have to record it, it takes you, like, ten minutes.

HS But you weren't making things at first; you were talking about what happened to make you shift from just talking about science subjects to making things in front of the camera?

JG I wanted to start making stuff right off the bat. But, like I told you, I didn't have any skills, because I was an engineer. We're good at designing, but not so good at making.

I was a broke college kid, so I didn't have any money. And a Chinese company contacted me, and they were like, "Oh, would you mind reviewing our printer? We'll send it for free and you can review it." And I was like, "Yeah, wait, are you sending me a printer for free? Oh my God, thank you!"

I got addicted to 3D printing immediately. And the first thing that came to mind was the Tesla turbine.

That was my first video that went viral. And when I say viral, I got like 100,000 views which, for me, was insane. That's how I got started. And then it took me a while to find my place on YouTube. Like, I'm the 3D printing jet engine Nikola Tesla guy. It took me a while to find it. But now I'm here.

HS Do you have a bit of competition from Colin Furze in the jet engine area?

JG I don't have competition because I love that guy!

I started YouTube because of him. I remember being in college and watching his videos. I was with him in San Francisco and he told me that his first camera was a Super 8 video camera.

That's crazy: I thought it was a GoPro. That's how long he's been on YouTube. I think he started his channel in 2006 – YouTube only started in 2005. And I've been watching him since forever. And he kind of has that same vibe that I do, in the sense that you can do it. Right?

It doesn't matter how hard it seems or how complicated. A turbojet engine? You can do it. You can do it with a roll of toilet paper and a turbocharger from a car. Just put it together with tape and it works. And I think that that is the vibe that I always tried to mimic, that everyone can do it.

HS And you're both obviously having a lot of fun?

JG That's the thing about it. It can become a little bit dangerous sometimes. And especially for him. I think I'm like a lightweight. There was actually a panel at Open Sauce, talking about dangerous people. And I didn't even make the panel because it filled up before I could attend the panel. I think it was Colin Furze, Styropyro, ElectroBOOM, The Backyard Scientist, and someone else.

But anyway, there were a lot of dangerous YouTubers there. There's an old video by Colin Furze in which he literally burns his entire arm. And he vlogs it because he goes to the hospital and he's like popping the blisters on his burns. Nowadays he doesn't really get hurt. As much.

HS OK, so if you were starting out today building your first jet engine, where would you start?

JG I would use metal because now I know enough about welding that it becomes much easier to make, for example, a pulse jet engine, with sheet steel.

At the time, I didn't know how to weld, so I decided to go another way. I found this Swedish YouTuber who developed this way of using carbon fibre and this weird refractory plaster that can take high temperatures. And he basically used 3D printing in that to make a pulse jet. And I was like, "That's my vibe; I can make it happen even without knowing how to weld". In that case, I did it; I just used the same method.

Nowadays, if I could do it again, I would use welding and sheet steel metal. But what I really want to do now is do a turbojet engine that is 100% metal 3D printed. A lot of people don't know about this, but metal 3D printing just became very affordable. Not the machine itself, but the service, so you can get metal parts 3D printed really cheap.

The first thing I tried to get printed was a wrench ring – a little spanner that's shaped to go around your finger like a ring. And that cost me $5. The sky's the limit now. I can do whatever I want.

HS Have you tried metal 3D printing for more functional parts?

JG I made a jet engine based on a Dyson hairdryer.

It felt like cheating. Normally when you make a jet engine, there's a lot involved.

What I did for combustion chambers a while ago was spot-welding, which is hard. It literally takes forever just to [make] the combustion chambers. And, with metal 3D printing, I literally just 3D-modelled it. I sent it to the company, and then it was done. It arrived back here, I assembled it, injected some hydrogen into it, and it was working.

OK, I'm lying. I had to do the injection system. But that's not that hard.

That's the thing with metal 3D printing. It makes stuff that ordinarily would be impossible to make by hand so easy. The machine does all the work, and we just need to enjoy the flames.

I got so much hate for that video, because I bought three Dyson hairdryers, and I destroyed two.

When I bought the hairdryers, it was around Mother's Day, and the girls in the shop were impressed, because they thought I was buying them for my mother or my sisters, or something like that. I was like, no, I'm gonna destroy them all.

They wanted to punch me in the face for that but, for me, it was like buying lumber. It was like something that I needed for the video.

And I remember when I was younger, I would love to break stuff apart to see how it worked. And my mother told me, you know, you're never gonna get a job destroying stuff.

I used to remove the motors from RC cars to make RC boats. Now, I basically destroy hairdryers to make a jet engine. So, it's almost the same thing.

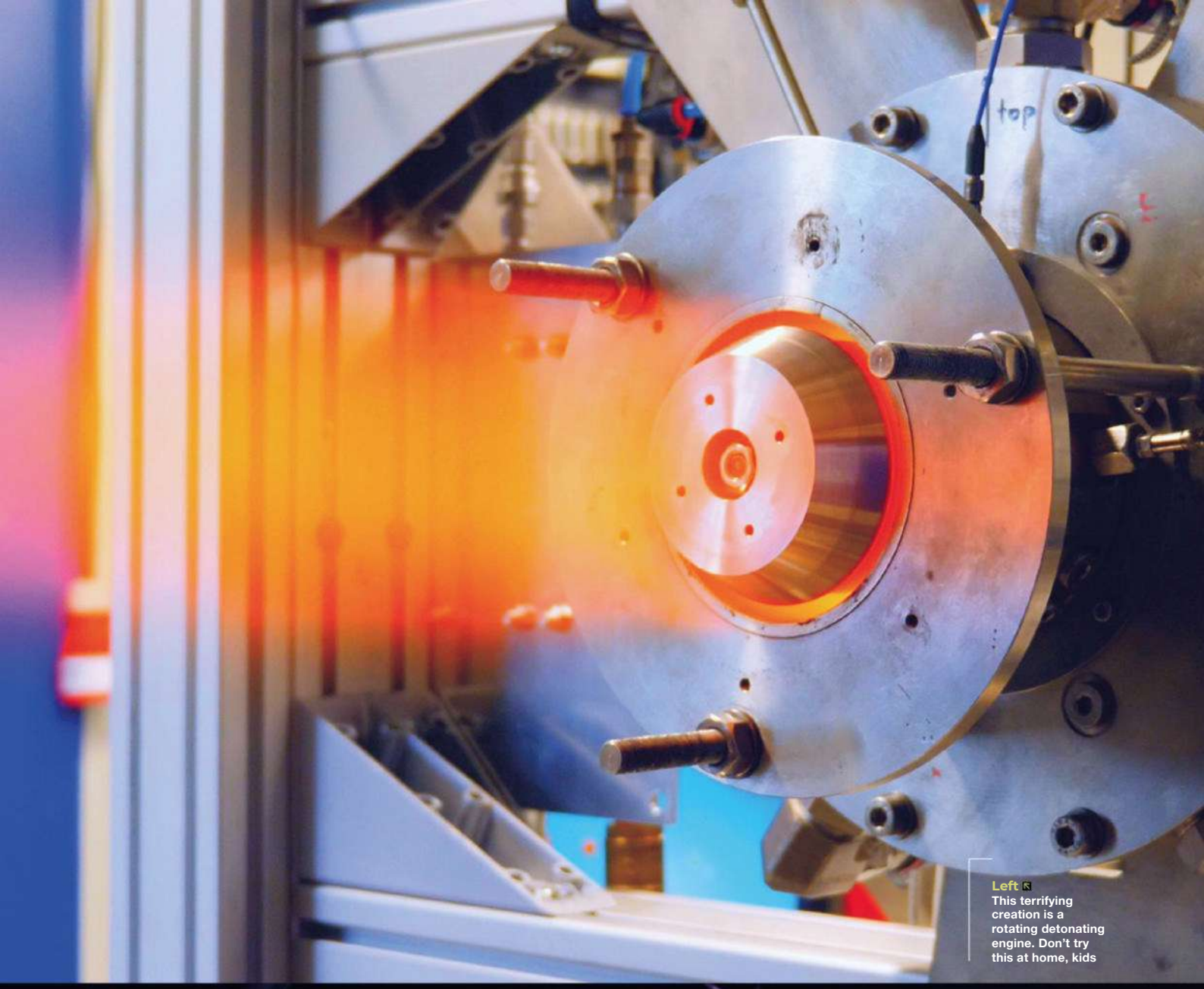
HS Do you have a favourite among your projects?

JG I think it would be the pulse jet, because the day I made the video was the first time I'd managed to get it to run. I was so happy, and you can tell on the video. I can't even hide it properly. I'd been trying to make my own pulse jet engine for like nine months, I failed and I failed. And I failed. Because starting a pulse jet engine, especially the valveless ones, is a form of art: you have to give it a little bit of compressed air, a little bit of butane… you hear that rumbling start to build up. And you have to give it more air and a little more butane until it starts and you can remove the air and it works just with the fuel source. It was so hard.

It's funny because I talked about this with Colin Furze, and he says exactly the same thing. It looks very easy on camera, but it's so hard. I built the engine the night before and woke up early on Saturday because I was really anxious to see it working. Thank God the camera was rolling. I was ecstatic. It was super-loud. My family all woke up. My mother was concerned because she thought I had killed myself. No other project has me so happy, because it was the first time I made a jet engine with my own hands, and using 3D printers.

HS What is it about jet engines that you think has captured your imagination?

JG There's a little bit of alchemy there too… you're taking a little bit of this chaos around you, and funnelling it into defying entropy. Jet engines are a very unnatural thing. When you look at them, you can tell that they are not supposed to exist in nature. But it's funny because in our core, we can't really explain or justify the things we like, or don't like. So I can't justify why I like jet engines so much. I think it's the raw power of it. Even more so rocket engines, because it's just, it's raw power, it's just flames and thrust. It's insane, it's scary, and I love them. ◘



Left
This terrifying creation is a rotating detonating engine. Don't try this at home, kids

Right
Joel's been experimenting with incorporating a 3D printed Tesla valve into a pulse jet engine

FEATURE

IN THE WORKSHOP: Tombola

Building a game of chance

By Ben Everard

Our local community garden held a summer event to raise funds, and in the style of fêtes, fairs, and fests across this fine island, a key part of the fundraising is the tombola. If you've not come across such a thing before, the basic idea is that you have a lot of tickets with numbers on them; they're folded up and put into a cylinder that can spin. You spin the cylinder around a few times, then open up a little door. A contestant then picks out a ticket. Depending on the number they pick out, you may or may not win a prize.

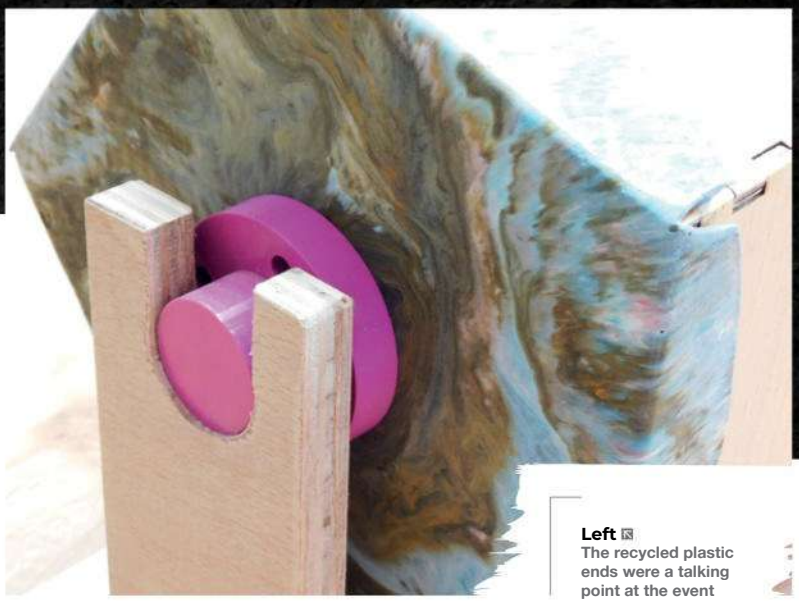
Below
I'm a bit disappointed that I didn't get the hinges working in time, but it worked well, and the event was a success

You can buy tombola machines, but we wanted to try and build one out of bits from our scraps pile. After all, there's no point in having a fundraising event if you have to raise funds to buy the things you need in order to raise funds.

The two main bits I pulled from the scraps pile were a plywood board and some discs of recycled PLA. Neither of these is particularly good for cylinders, so we had to compromise and make a pentagonal prism. The wood was used to make the five sides, and the PLA was used for the ends. The ever-useful Boxes.py tool (located at **festi.info/boxes.py**) can generate an SVG file for a pentagonal prism that can easily be laser-cut from the plywood. We could also have laser-cut the ends from the PLA discs, but we wanted to try heat-forming them.

The idea was simple – place a flat sheet on the end, hit each side with a heat gun, fold it down, and then drill through and bolt it into place. Since this was the first time trying to work with the PLA sheet like this, we weren't quite sure how it would go. We've made bowls with it before, but this time we needed to be more precise.

Yet again, we were pleasantly surprised by how easy it was to work with softened PLA. Since it becomes workable at a low temperature, it's easy to work with, and it's easy to vary the hardness from a bit bendy to almost completely flaccid depending on



Left
The recycled plastic ends were a talking point at the event

We could quite easily have made a smaller one

Above
PLA sheet is easy to work with hand tools. We bent this with a heat gun and snipped the corners with scissors

Left
The 3D-printed axles made it easy to mount the tombola on the frame

how hot you get it. It's also easy to snip into shape with a regular pair of scissors. We used a block of wood to stop the hot air from reaching parts of the plastic that we didn't want to deform, and that also worked well to isolate the heat and make it malleable only where we wanted it to be.

The cylinder was now made, but we needed a way to spin it, and this seemed like a good job for 3D printing. We made a simple stub-axle with holes to bolt it onto the PLA sheets. The final thing was just to build a base from some other scraps of wood.

This gave us a spinning cylinder, but we needed a hole which we could use to draw the tickets out of. Because of the size of each side, this had to span two sides in order to be big enough. We cut out the opening, then bent another piece of PLA to the right shape to fit over the gap.

In retrospect, it's a ridiculously large tombola – we could quite easily have made a smaller one, but it'll keep us going for quite a few fundraising events yet. What's more, we've learned more about how to work with the recycled PLA sheet made from 3D printing waste. We're increasingly finding that this is a really useful material. ◘

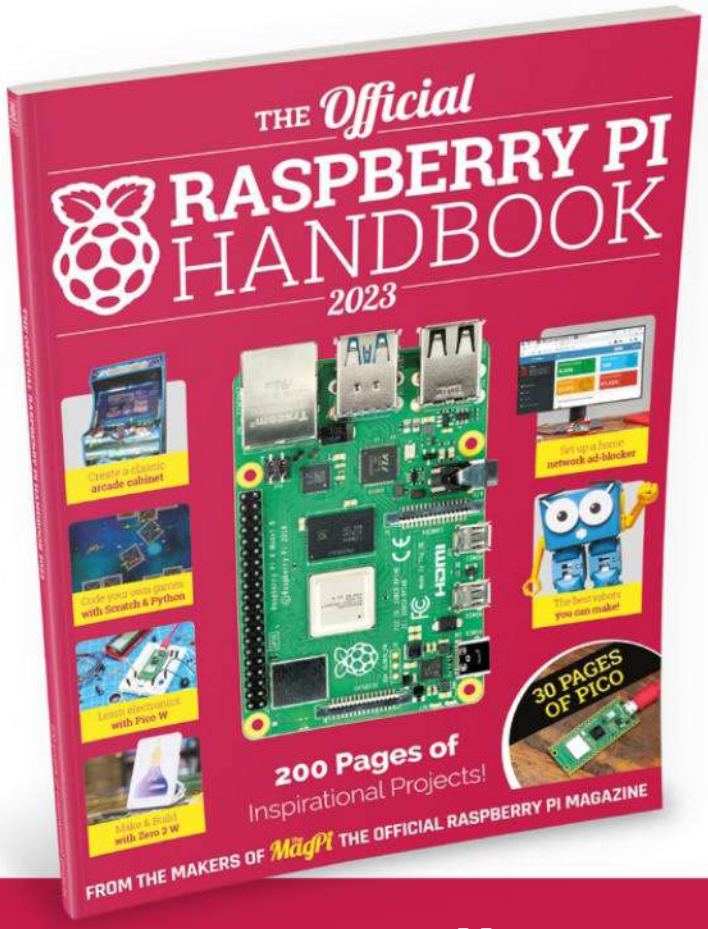
Laser-cutting PLA update

Last issue, we looked at jewellery making with this waste PLA sheet. Perhaps the most time-consuming part of the process was cleaning up the parts after laser-cutting. It took quite a bit of fiddly sanding for each part. After some experimentation, we found that sandblasting is an excellent and quick way of tidying up the bits before flame-polishing them. This speeds up the process of creating decorative items from the PLA sheets with swirling colours.

THE Official RASPBERRY PI HANDBOOK 2023

200 PAGES OF RASPBERRY PI

- QuickStart guide to setting up your Raspberry Pi computer
- Updated with Raspberry Pi Pico and all the latest kit
- The very best projects built by your Raspberry Pi community
- Discover incredible kit and tutorials for your projects



Buy online: magpi.cc/store

HackSpace
TECHNOLOGY IN YOUR HANDS

FORGE

HACK | MAKE | BUILD | CREATE

Improve your skills, learn something new, or just have fun tinkering – we hope you enjoy these hand-picked projects

Designing an RP2040 board using KiCad

Build a custom microcontroller board and get it assembled

Jo Hinchliffe

@concreted0g

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!

In the previous parts of this series, we've worked through the basics of making PCB boards from schematic to board layout, and we've learned a variety of skills and approaches along the way. We built on this by exploring how to use KiCad and a PCB assembly (PCBA) service to not only have the PCB manufactured, but to also be populated with components and supplied to us fully assembled (**Figure 1**). If you've worked through the earlier parts, we now have enough skill and knowledge to tackle a more extensive project like creating an RP2040-based board.

RP2040 is a great target for PCB projects that will be assembled using industrial approaches. The bolder amongst us might successfully be able to solder up the QFN (Quad Flat No-leads) 56 package at home, but it's at the edge of where PCBA starts to make a lot of sense. Many commercially available boards that use the RP2040 are manufactured using PCBs with four or more layers. Of course, it's possible to do this in KiCad, but for many of us, four-layer boards can be difficult to debug or correct if something goes wrong. Fortunately, for simpler boards, two layers is enough to get our microcontroller running and break out the features we need.

Figure 2 The excellent Hardware design with RP2040 documentation from Raspberry Pi

When the RP2040 was released, so too was a stack of excellent documentation, including the very readable 'Hardware design with RP2040' (**Figure 2**). The PDF is available to download from **hsmag.cc/hardware_design_rp2040**.

This document shows an example of a minimal RP2040-based board, a two-layer design, and then describes various aspects of the design and considerations. There is even a KiCad project file for the design. It's a great idea to download the project file and take a look around it (**Figure 3**). In this article, we are going to replicate this board design, but we will start from a blank project rather than use the one supplied.

Starting from scratch means that it's easier to adapt this project to your own needs. We'll also tweak the project because currently, JLCPCB doesn't supply some of the exact components used on the Raspberry Pi example. The Raspberry Pi example was built in a previous version of KiCad and uses in-house schematic symbols for the RP2040. Since then, the RP2040 has become a standard library component within KiCad, but some of the connections on the RP2040 schematic symbol are already made at a symbol level (like common power pin connections), so it makes sense to use the built-in KiCad symbols and footprints. With that all said, we aren't going to step through every stage of designing this board as we've learnt the approach in the previous parts of the series, so this article

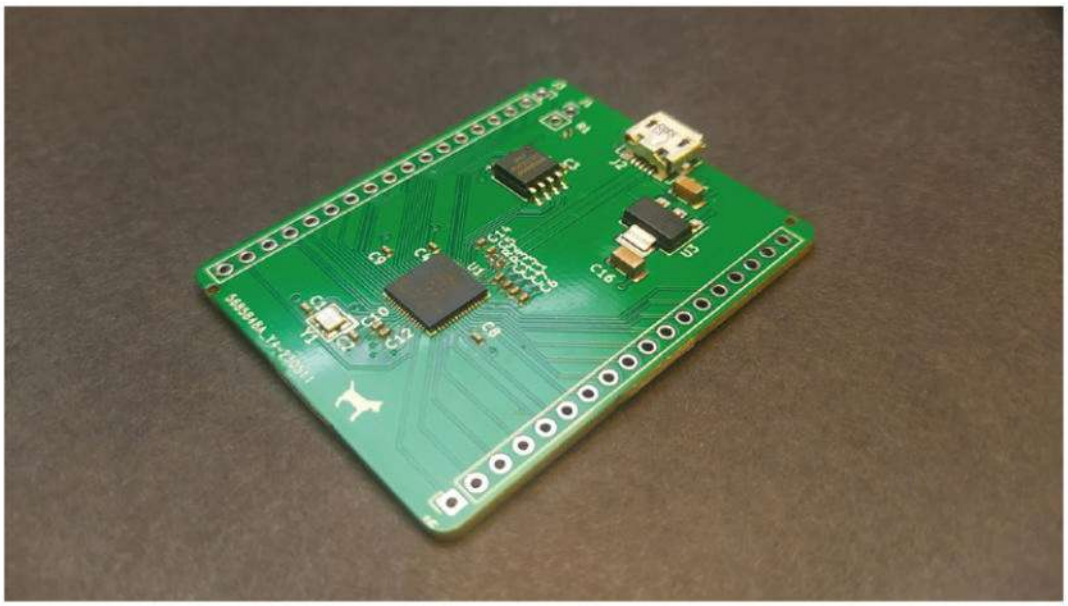


Figure 1
A fully assembled and functional RP2040-based board

looks at things we consider specific to creating an RP2040 board.

We've tended to emulate both the schematic layout and the PCB layout of the minimal design example and our project. Partly this is because the PCB layout has neatly solved a lot of the layout complexity, but also it allowed us to compare the project as we worked through building our own. With the USB symbol (and footprint) sourced, we added the RP2040 component from the KiCad

> **Starting from scratch means that it's easier** to adapt this project to your own needs

library. We added resistors and a pair of labels to connect the D+ and D- to the correct pins on the RP2040. The documentation states that we need to create the traces for these connections with accurate dimensions and clearances. We'll achieve this by assigning a custom net class to the connection, or 'net', so that when we draw these traces in the PCB Editor, they should be created correctly.

In the Schematic Setup dialog in the Schematic Editor, click the + button in the uppermost Net Classes window, add a new net class, and give it a name. Note that local net class names within a project should start with a '/' – we went with /USB_lines. Select a wire segment that connects the RP2040 D+ or D- pin out to the USB_D+ or USB_D- label that we added, and then right-click. Select Assign Netclass from the drop-down menu. In the Add Netclass Assignment dialog box, you should see the selected label USB_D+ and, to the right of it, a drop-down menu to select the net class – this is currently 'Default', but if you click down, you

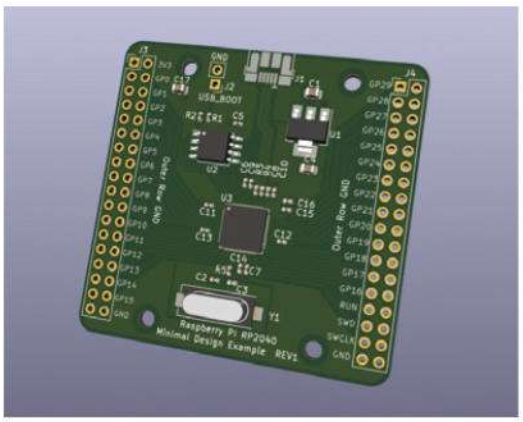


Figure 3
The Hardware design with RP2040 documentation also includes a KiCad example project

QUICK TIP

Reading through the Hardware design with RP2040 PDF a couple of times is beneficial to laying out an RP2040-based project!

QUICK TIP

Do take care when creating custom footprints to ensure that the component pin number is compatible with your schematic symbol and vice versa.

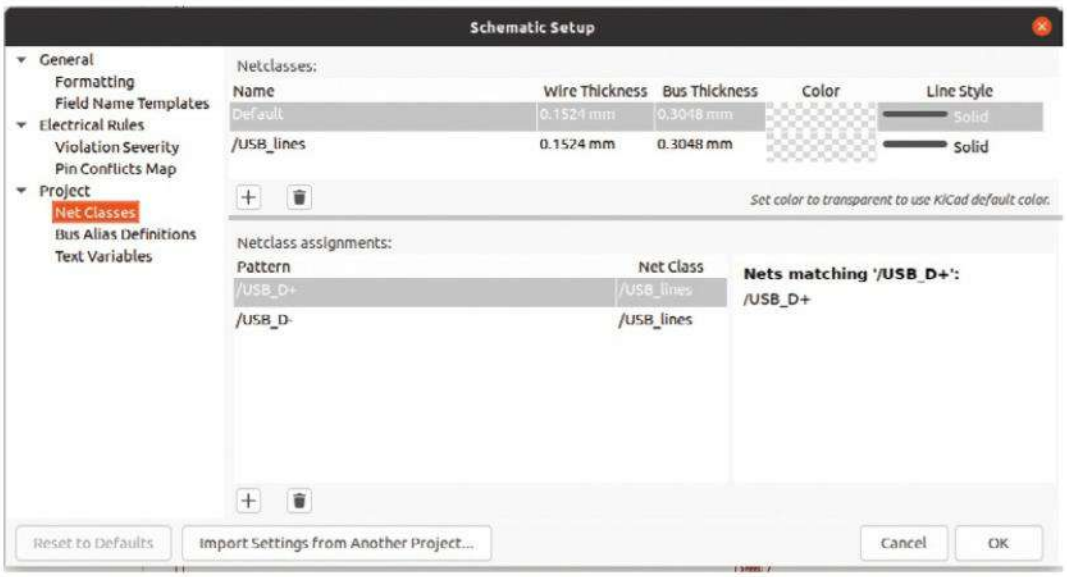
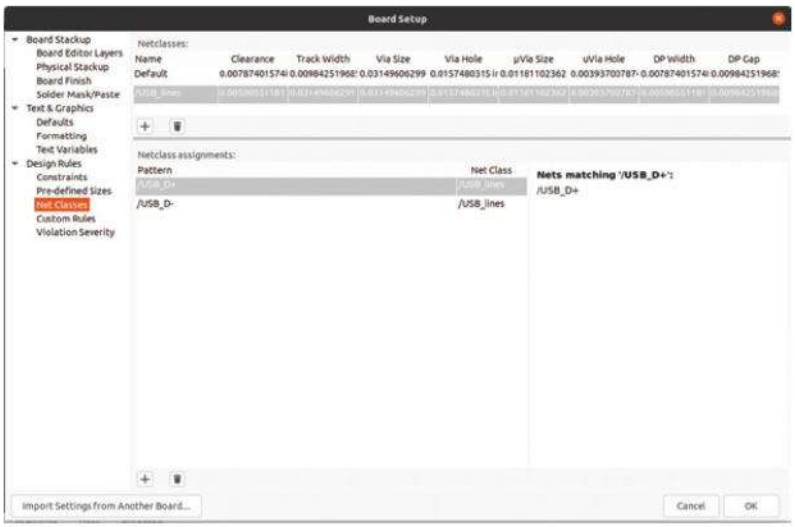


Figure 4 Setting up and assigning a net class

should be able to see the 'USB_lines' net class that we added earlier. Select this, then close the dialog box. Repeat this for the USB_D- label (**Figure 4**).

When you get to opening the PCB Editor, you can use the Board Setup tool (in the same positions as the Schematic Setup tool) to adjust the net class variables. This includes the track width, track clearance, via size, and more (**Figure 5**). We are predominantly interested in the track width and the track clearance to create the track lines that we need for the USB lines – this information on track geometry was taken directly from the Hardware Design with RP2040 documentation (page 10).

Figure 5 Setting the net class variables for our USB lines in the Board Setup dialog in the PCB Editor



We laid out the power regulator and found a similar device to the minimal design example; however, the JLC component, C26537, had an extra pin and was in the SOT-223-3 package. We have a matching KiCad library footprint, but it was easier to just make a quick custom schematic symbol to ensure that the pin numbering was correct and matching (**Figure 6**).

One area that challenged us was, when looking up the components that the minimal design example project used for the crystal, we found no similar device available on the JLCPCB component library. Using a consummate hardware hacker's approach, we researched what other open-source RP2040-based designs used, making sure to limit the list to projects we knew worked well. Having used Solder Party's Stamp in an earlier part of this series, we checked out its design. It looked simple and straightforward with just two 12pF capacitors, and on checking JLCPCB, the crystal part was available as part number C521567. We again used the EasyEDA conversion website (see box, opposite) to create a custom footprint (**Figure 7**, overleaf).

Laying out the decoupling capacitors in the schematic is straightforward, but we've taken advantage of the KiCad text tool to add notes, occasionally acting as reminders for important information. In the Hardware design with RP2040 documentation, for example, it shows that the 1uF capacitors should be placed close to pins 44 and 45 on the RP2040, so it makes sense to add a schematic note as a reminder (**Figure 8**, overleaf).

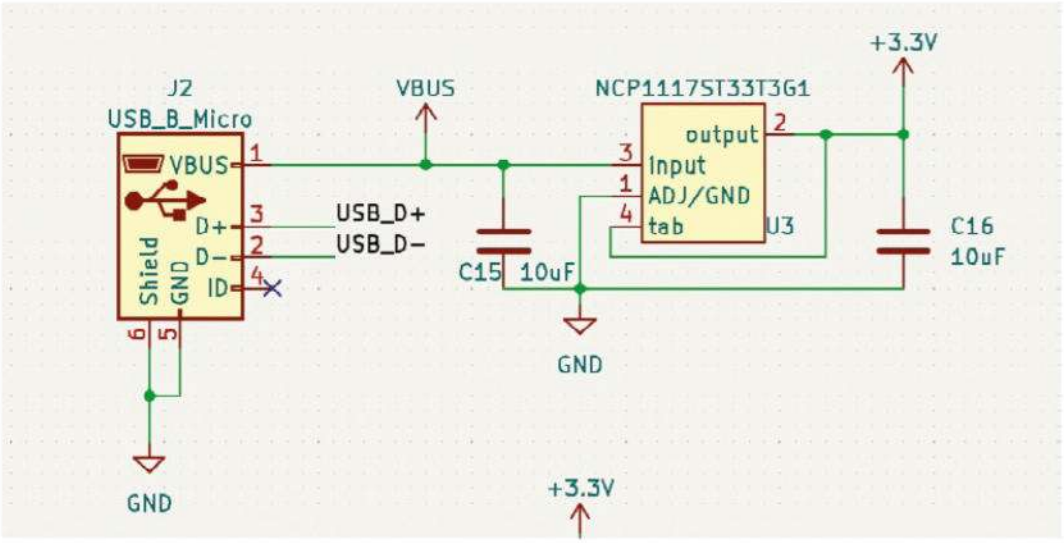
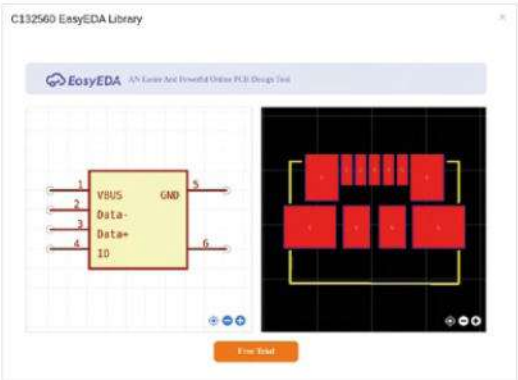


Figure 6
Setting up the power regulator is simple once the target JLCPCB components are identified

The flash memory chip used in the minimal design example is available at JLCPCB and, as such, we went with the same design. In the Hardware design with RP2040 documentation, they have added a footprint for an optional pull-up resistor but found, with this chip, that it wasn't needed, so we omitted that part of the design. The important thing about the flash chip and the associated traces on the PCB is that it all sits over a continuous ground plane. This adds some head-scratching with the lay up of a two-layer board, but again, use our project or the Raspberry Pi project as an example of the routing.

As we aren't aiming for any particular use case with this example, we simply broke out all the pins to a pair of headers that we will place on each side

> **We simply broke out all the pins to a pair of headers** that we will place on each side of the board



Above
LCSC provide EasyEDA symbols for most of their components so you have to convert these before you can use them in KiCAD

EASILY CONVERTED

When designing for JLCPCB, even at the schematic level, you need to be thinking about what component and footprint you will be using. For our RP2040 board, we wanted to use a surface-mount micro USB-B socket where all the USB chassis and ground points are on the upper layer – this means we could keep all the assembly as SMD components. As such, apart from the pin headers/sockets, the units would be fully assembled by JLCPCB.

Looking through the LCSC component library, we opted for the C132560 component. JLCPCB often supplies schematic and footprint symbols for EasyEDA, and, in the case of the component footprint, we can use a handy online tool to convert it into a KiCad component footprint and add it to our libraries (we covered working with custom libraries and components in the third part of this series – **hsmag.cc/issue68**).

To make use of this converter, you'll need to set up an EasyEDA account (**easyeda.com**). Click the link that says 'PCB Footprint or Symbol' on the component page on the JLCPCB library website, then click the Free Trial button underneath the pop-up window that shows the component symbol and footprint. After setting up a login, you should see a browser-based EasyEDA project with just the footprint of the component loaded. In the browser, go to File > Export, and then select EasyEDA … from the list. This should then download a small JSON file. Navigate to **hsmag.cc/EasyEDA2KiCad**, click the Load EasyEDA File button, and then upload the JSON file. The website will automatically convert the EasyEDA project and will then download a KiCad footprint file to your Downloads folder. As a side note, we also used this approach for the crystal package we used on the board design.

Sadly there is no conversion available for the schematic symbols; however, with some careful checking of pin numbers to ensure compatibility, we found a built-in KiCad USB schematic symbol that worked well with the footprint. Again, though, if you've worked through all parts of this series, you should be happy to create your own custom symbol.

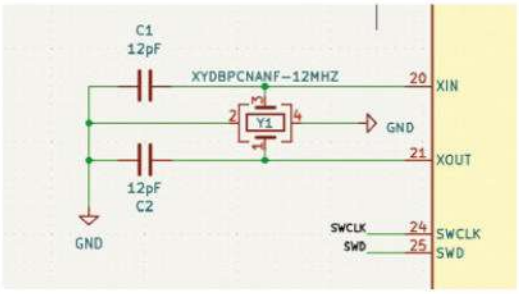


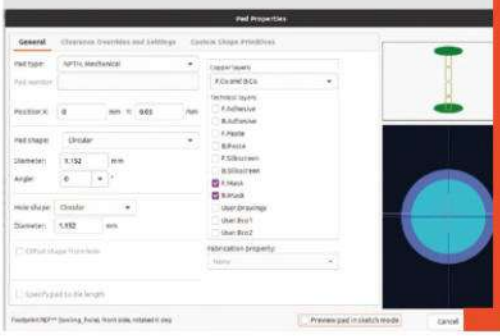
Figure 7
Due to JLCPCB not stocking the crystal that the RP2040 documentation recommends, we needed to rethink this part of the circuit

TOOLING HOLES

When ordering assembled PCBs from JLCPCB services, you might need to consider tooling holes. These are small holes placed in your design layout that JLCPCB machines use to locate and hold the boards when they are being assembled. You can add these yourself, or you can omit them, knowing that JLCPCB will place them for you. It's fair to say that JLCPCB engineers will place the holes sensibly and won't plonk one through a trace or in the middle of a component footprint; however, you might want to manually add them into your design so that you decide where they are finally placed.

The rules are that a minimum of two, preferably three, holes should be placed in the PCB design, and they should be placed at opposite corners – as far apart as they can practically be. The holes should be 1.152 mm diameter circular non-plated holes with a 0.148 mm solder mask expansion. The simplest thing we found to do for projects where we want to add the tooling holes is to create a custom component in our KiCad libraries that we can drag into and place in our design.

To make a tooling hole component footprint in the Footprint Editor, you need to create a new footprint and then add a single pad. Selecting the pad, click the **E** key to edit and enter the Pad Properties dialog. On the first 'General' tab, set the pad as 'NPTH, Mechanical' in the pad type – NPTH stands for 'non-plated through-hole'. Staying on the General tab, make sure the pad shape is set to circular and the diameter is 1.152 mm. Finally, click onto the second tab in the Pad Properties dialog called 'Clearance Overrides and Settings'. On this tab, set the solder mask expansion to 0.148 mm. Save this as a footprint and you can place them when needed into JLCPCB-oriented designs.



of the board. With the schematic largely complete, we moved over to the PCB Editor to begin the layout and routing.

One interesting aspect of the board design, that we haven't looked at yet, is that it has numerous different voltages which each have their own copper flood zones. Making these is similar to how you make any other copper flood, as we have done in previous boards, but you need to set a priority value so that the different floods know how to flood separately. So on the top layer of our board there is a general 3V3 flood, a flood connected to the VBUS, which is the 5 V input from the USB to the voltage regulator, and there is a small 1V1 zone inside the footprint of the RP2040. Notice in **Figure 9** that we have assigned the 1V1 flooded area priority 2, the VBUS area priority 1, and the general 3V3 priority 0. This essentially shows that they are separate areas.

Components-wise, we decided that for the resistors and the majority of the decoupling capacitors, we would go for 0402 packages. Again, we wouldn't make this choice if we were planning to assemble this board by hand, but with the assembly engineers and robots doing this fine work, we might as well use the tiny packages. It's less common for

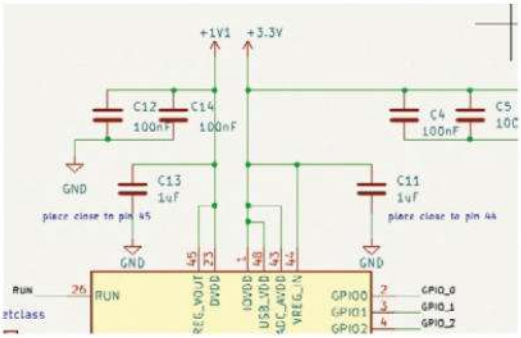


Figure 8
Laying out the decoupling capacitors in the Schematic Editor is largely straightforward

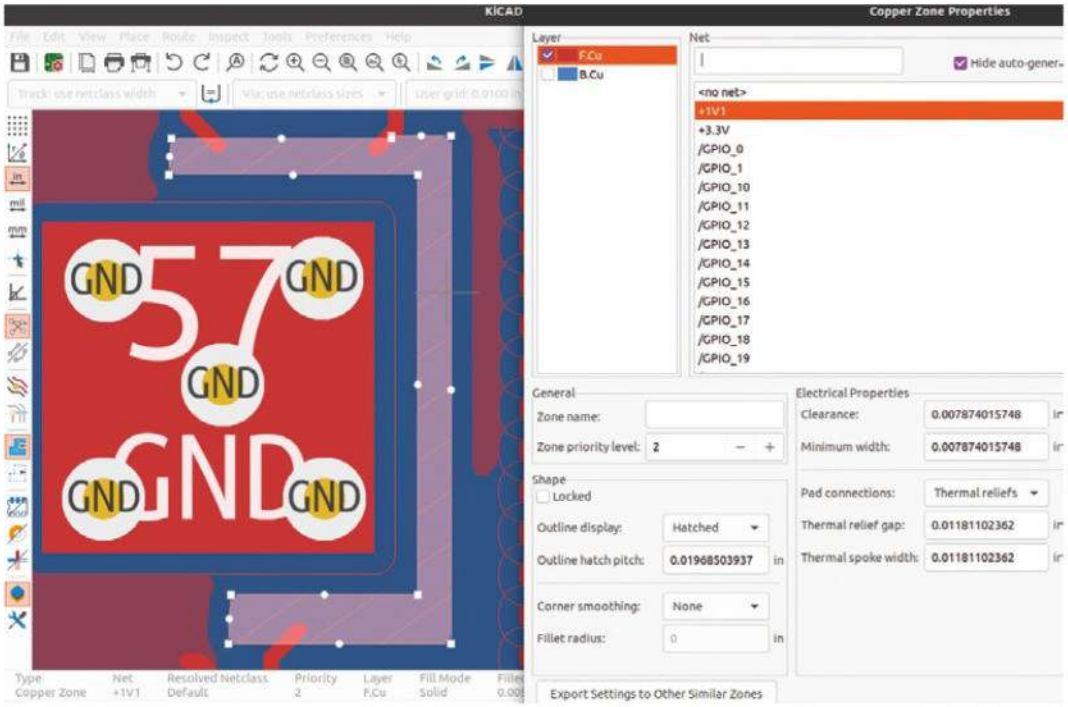
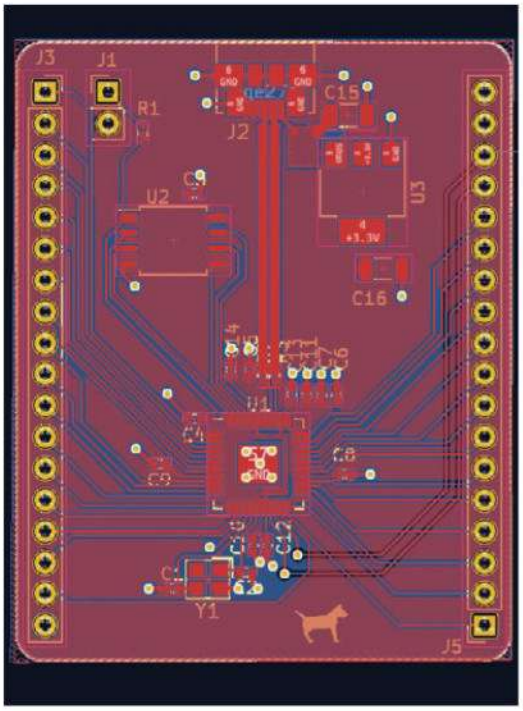


Figure 9
Setting up different priorities to allow different zones to coexist

> **With the assembly engineers and robots doing** this fine work, we might as well use the tiny packages

the very tiny packages of capacitor to be of accurate value as they increase in capacitance. So, although JLCPCB does offer components that claim 10uF in 0402 packages, we went with a more common larger 1206 variant.

We've largely covered everything specific to this project in this article. If you've worked through the previous parts of this series, you'll be familiar with all the other aspects of the process. It's definitely worth working through some smaller projects (like the small H-Bridge design in the last part of the series – **hsmag.cc/issue69**) to get used to the JLCPCB-specific processes. We'd also reiterate that reading Hardware design with RP2040 a couple of times before tinkering with an RP2040 board is time well spent. Finally, be warned, having fully assembled and hopefully functional boards delivered to your door is highly addictive! The KiCad files for this project can be found at **hsmag.cc/issue70**.



Above
Our completed layout in the PCB Editor

TUTORIAL

Part 04

Build a NeoPixel LED colour changer with dials

Learn about analogue inputs on the Raspberry Pi Pico using potentiometers. Use the value of the dials to change the colour of RGB LED NeoPixels

MAKER

Stewart Watkiss

Also known as Penguin Tutor. Maker and YouTuber that loves all things Raspberry Pi and Pico. Author of *Learn Electronics with Raspberry Pi*.

penguintutor.com

twitter.com/stewartwatkiss

In the world of computers, data is stored as digital values with electronic switches that turn on and off. The real world is analogue, with different levels of brightness throughout the day or different temperatures. This tutorial will shows one of the ways of taking analogue values from the real world and converting them to a digital signal that a Pico microcontroller can understand.

Knowing the value of the analogue inputs, these can then be converted into colour values which are used to light up NeoPixel LEDs.

You'll Need

- 2 × half-size breadboards **magpi.cc/breadboardhalf**
- 3 × 10 kΩ potentiometers **magpi.cc/10kpot**
- 2 × NeoPixels **magpi.cc/smartneo**

01 Potentiometers

Potentiometers are a type of variable resistor, a resistor that changes in value. They have three terminals and often have a shaft that can be turned to change the resistance. You can also get miniature potentiometers which can be adjusted using a screwdriver.

Inside the potentiometer there is a resistive track with a slider that moves along it. As the slider moves towards terminal 1, the resistance between terminal 1 and terminal 2 decreases, and the resistance between terminal 2 and terminal 3 increases. A photo and schematic symbol are shown in **Figure 3** (overleaf).

02 Voltage divider

A common way of using a potentiometer is as a voltage divider. Connecting the voltage divider across the power supply, the output voltage will change as you turn the shaft. To understand how this works, we can imagine the potentiometer as two separate resistors, as shown in **Figure 2**. One resistor goes from terminal 1 to terminal 2 (R1), the other from terminal 2 to terminal 3 (R2). Connect terminal 1 to ground and terminal 3 to the supply voltage. The voltage output is proportional to the ratio between R1 and R2. For a 3.3V power supply, with the wiper towards terminal 1, the output will be 0V; with the wiper towards terminal 2, the output will be 3.3V; and halfway between, the output will be 1.65V. This provides an analogue output voltage.

03 Analogue to digital converter

If we want to use the analogue voltage as an input to a microcontroller, it needs to go through an analogue-to-digital converter (ADC). Fortunately, there are three inputs on the Pico which have built-in analogue-to-digital converters. These are physical pin 31 (ADC0), pin 32 (ADC1), and pin 34 (ADC2). These pins sample the voltage and provide a digital value for it. In

MicroPython this is represented as a value between 0 and 65,536 (although the resolution of the ADC is 12 bits and not 16 bits as the range suggests).

04 Sampling the analogue value

To see this in action, connect the potentiometer to the Pico between pin 33 (GND) and pin 36 (3.3V output). Then connect the centre wiper pin to pin 31 of the Pico (ADC0). With just three lines of MicroPython code, you can query the value of the ADC port.

```
from machine import ADC, Pin
adc = ADC(Pin(26))
print (str(adc))
```

Running this will show a number representing the value at the pin. It's unlikely it will cover the full range, but you should be able to get a reasonable range starting at about 300 upwards.

If you divide the value received by 19,859 (roughly 65,535÷3.3), you should get a fairly accurate representation of the voltage.

05 Adding more potentiometers

The colours we see on a computer screen are created using three primary colours: red, green, and blue. These are referred to as 'RGB'. To be able to represent different colours, we use three potentiometers, one for each of these colours. The potentiometers used for this can be plugged into a breadboard, but there isn't enough space on a single breadboard. Breadboards are designed to be joined, so you can use two breadboards connected (as in the **Figure 1** wiring diagram) or one full-size breadboard.

Alternatively, you can use potentiometers with wires connected to them, in which case they can be connected to the appropriate pin using just one breadboard.

06 Addressable RGB LEDs

To represent the colour output, we will be using addressable RGB LEDs. These are also known by their model number, such as WS2182B, or Adafruit refers to them as NeoPixels. These are often found on long strips, or made into rings, but you can also get them as individual pixels which can be connected to a breadboard. You can

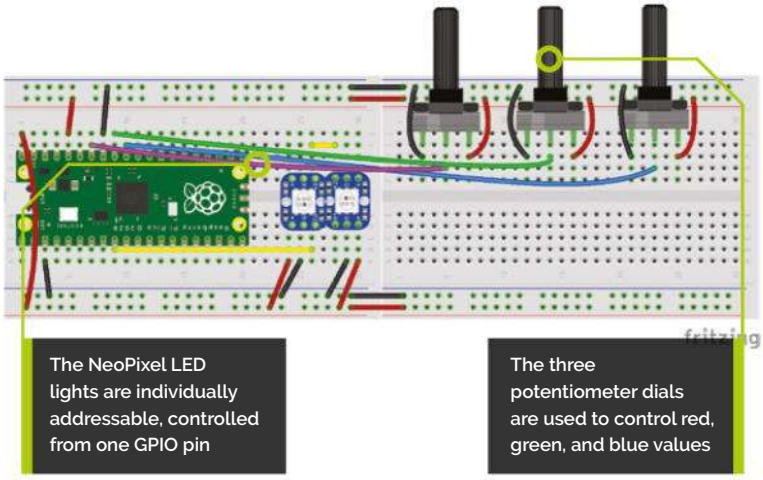


Figure 1 The breadboard wiring diagram for the circuit

> We will be using addressable RGB LEDs

individually set the colour of each of the individual LEDs, but this example uses just two LEDs which will be set to the same colour.

To control the NeoPixels, you need to set the RGB colours, providing each value in the range 0 to 255. This can be achieved by dividing the ADC output by 257.

07 How addressable RGB LEDs work

Addressable RGB LEDs are the combination of multicolour LED (with elements for red, green, and blue) and an integrated circuit used to determine the colour. The LED is often mounted directly on top of the integrated circuit (IC), making these appear as a single unit. The microcontroller sends a long string of values

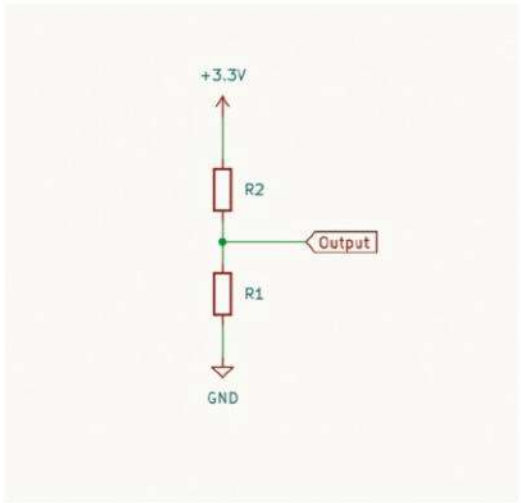


Figure 2 The potentiometer is connected as a voltage divider, shown here as two resistors. The values of R1 and R2 will depend on the position of the wiper

THE MAGPI

This tutorial is from in The MagPi, the official Raspberry Pi magazine. Each issue includes a huge variety of projects, tutorials, tips and tricks to help you get the most out of your Raspberry Pi. Find out more at **magpi.cc**

Figure 3 Schematic diagram and photo of a potentiometer. This type has right-angle pins that can be inserted into a breadboard

representing the desired colour for each LED. The IC reads the first value for setting its own LED and then passes the rest of the colour values along to the next LED. The microcontroller needs to generate precise timing for sending the values.

Top Tip

Other variable resistors

Swap the potentiometers for other variable resistors to detect the environment. A light-dependent resistor (LDR) with a 470 Ω fixed resistor is a good starting point.

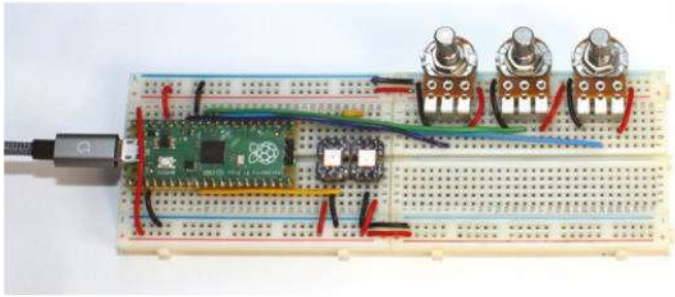
08 Wiring up the NeoPixels

The LEDs shown in **Figure 4** are Adafruit breadboard NeoPixels. The input of the first NeoPixel (labelled 'In') needs to be connected to an output pin of the microcontroller – in this case, Pico's GPIO 6 (physical pin 9). The output from the first LED (labelled O) then needs to go the input of the second LED. To make the wiring easier, you can alternate the direction in which the NeoPixels are mounted so that the output of the first is on the same side of the breadboard as the input to the second NeoPixel.

09 Powering the NeoPixels

The potentiometers are connected to the 3.3V output of Pico (pin 36). This is important, as exceeding that could damage the input pins on Pico. To provide normal brightness levels the NeoPixels needs to be powered with 5V, so they need to be connected to VSYS (pin 40) on Pico.

The data input for the NeoPixels should ideally be 5V but, using Pico's GPIO pins, it will be 3.3V. When the NeoPixels are physically close to Pico, that works fine, but if they were positioned further away then it may be necessary to buffer the signal first.



Two breadboards have been joined to provide space for the three potentiometers to be mounted. The top power rail is 3.3V for the potentiometers, the bottom power rail is 5V for the NeoPixels

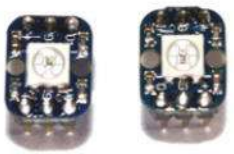


Figure 4 These NeoPixels can be inserted in a breadboard. There are three different coloured LEDs in the centre. The integrated circuit is hidden underneath the white square

10 Controlling the NeoPixels

To control the NeoPixels from your Pico, the signal needs precise timings. We don't need to get involved in the details, as the hard work of generating the correct signals is already included in a Raspberry Pi example program that can be found at **magpi.cc/neopixelringgit**.

The supplied code uses the Programmable Input Output (PIO) feature included in Pico. To use it, you just need to copy the appropriate functions into your own code, which is what has been done for the supplied code in **neopixel_rgb.py**.

11 How the code works

The first part of the code imports the required libraries and sets the values for the pin and number of LEDs. It then defines the three ADC inputs from the potentiometers.

The next section, marked with the comment block, is taken from the example code mentioned previously.

The code to read the values from the potentiometers and to then set the colours of the NeoPixels is contained within the `while True` loop. The three values are read separately and divided by 257 to convert them to 8-bit values. This is then combined into a tuple which is used to fill the LEDs with that colour. The `pixels_show` function is used to send the new values to the NeoPixels.

12 Other uses

The potentiometers could be changed for other devices that can measure the environment. This includes light-dependent resistors whose value changes based on the amount of light, or thermistors whose value changes based on the temperature (**Figure 5**). These would be wired as a voltage divider, replacing R1 with the variable resistor and R2 with a fixed resistor.

You could add more NeoPixels and set their values individually using the `pixels_set` function. You will need an external power supply when adding more NeoPixels.

neopixel_rgb.py

DOWNLOAD THE FULL CODE:
magpi.cc/neopixelrgbpy

Language: **MicroPython**

```
# Set RGB values based on analogue inputs
import array, time
from machine import ADC, Pin
import rp2

# Configure the number of WS2812 LEDs.
NUM_LEDS = 2
PIN_NUM = 6
brightness = 0.2

red_adc = ADC(Pin(28))
green_adc = ADC(Pin(27))
blue_adc = ADC(Pin(26))

@rp2.asm_pio(sideset_init=rp2.PIO.OUT_LOW, out_shiftdir=rp2.PIO.SHIFT_LEFT, autopull=True, pull_thresh=24)
def ws2812():
    T1 = 2
    T2 = 5
    T3 = 3
    wrap_target()
    label("bitloop")
    out(x, 1)               .side(0)    [T3 - 1]
    jmp(not_x, "do_zero")   .side(1)    [T1 - 1]
    jmp("bitloop")          .side(1)    [T2 - 1]
    label("do_zero")
    nop()                   .side(0)    [T2 - 1]
    wrap()

# Create the StateMachine with the ws2812 program, outputting on pin
sm = rp2.StateMachine(0, ws2812, freq=8_000_000, sideset_base=Pin(PIN_NUM))
# Start the StateMachine, it will wait for data on its FIFO.
sm.active(1)
# Display a pattern on the LEDs via an array of LED RGB values.
ar = array.array("I", [0 for _ in range(NUM_LEDS)])

##########################################################################
def pixels_show():
    dimmer_ar = array.array("I", [0 for _ in range(NUM_LEDS)])
    for i,c in enumerate(ar):
        r = int(((c >> 8) & 0xFF) * brightness)
        g = int(((c >> 16) & 0xFF) * brightness)
        b = int((c & 0xFF) * brightness)
        dimmer_ar[i] = (g<<16) + (r<<8) + b
    sm.put(dimmer_ar, 8)
    time.sleep_ms(10)

def pixels_set(i, color):
    ar[i] = (color[1]<<16) + (color[0]<<8) + color[2]

def pixels_fill(color):
    for i in range(len(ar)):
        pixels_set(i, color)

while True:
    red = red_adc.read_u16() / 257
    green = green_adc.read_u16() / 257
    blue = blue_adc.read_u16() / 257

    # Get a single value
    color_value = (int(red), int(green), int(blue))

    pixels_fill(color_value)
    pixels_show()
    time.sleep(0.25)
```

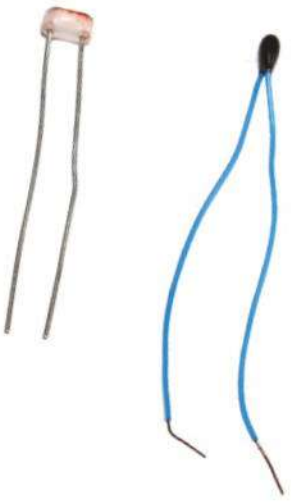
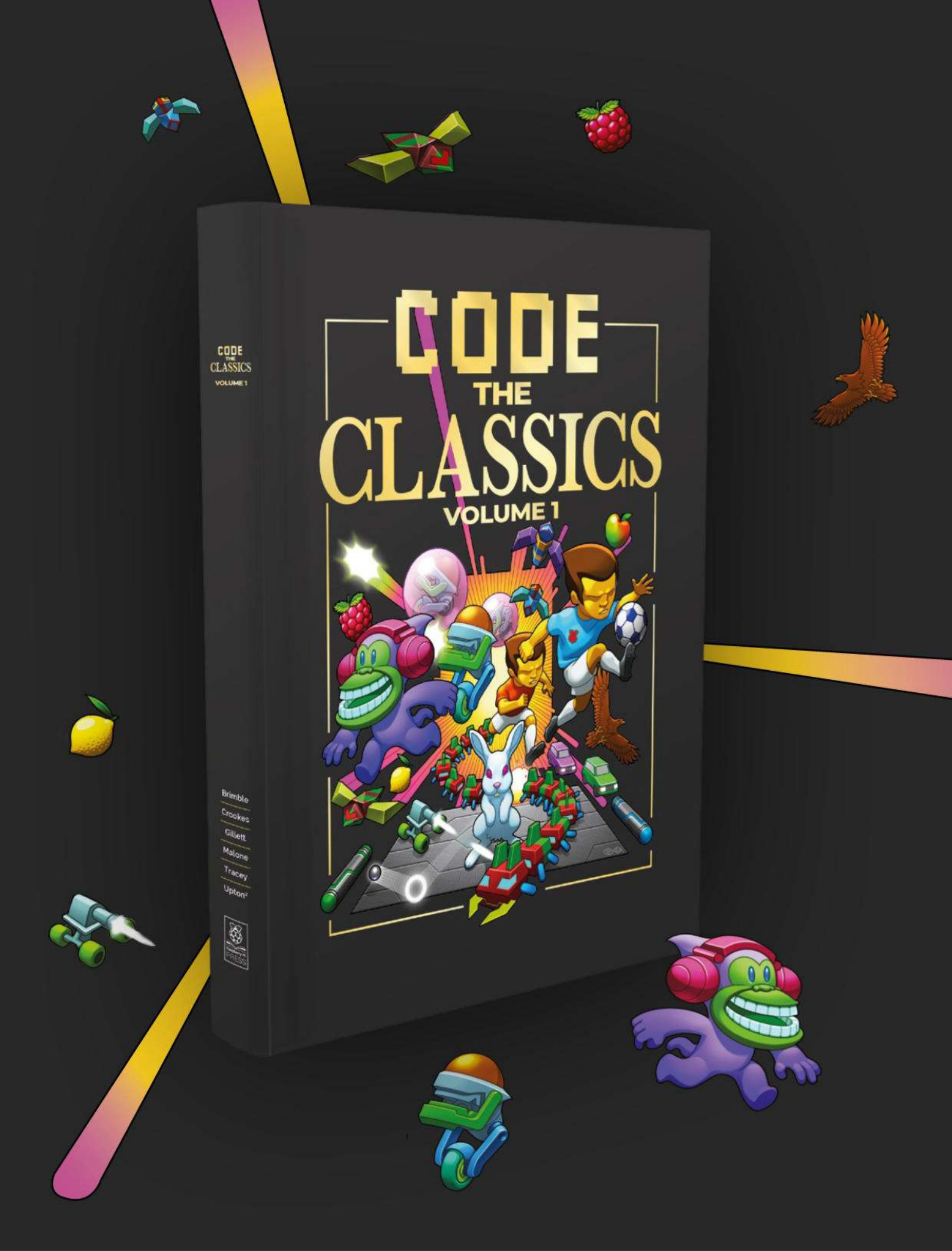


Figure 5 The light-dependant resistor and thermistor can be used to measure analogue light and temperature levels


CODE
THE
CLASSICS
VOLUME 1
Brimble
Crookes
Gillett
Malone
Tracey
Upton²

TUTORIAL

Blinking lights

Cinematography with 96 LEDs

Ben Everard

@ben_everard

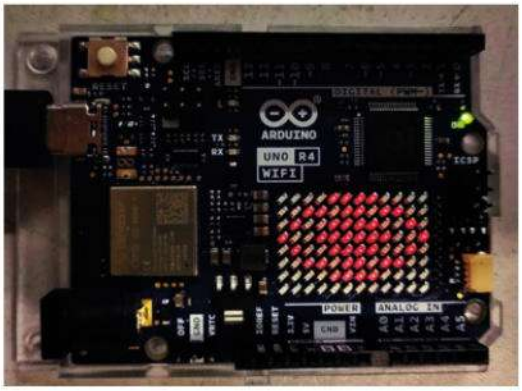
Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

The Arduino UNO R4 brings a whole bunch of new features to the Arduino UNO ecosystem, including a switch of the processing core at the heart. We looked at the changes in more detail in the last issue. This issue, we want to focus on just one of the features: the LED matrix on the WiFi variant of the board.

On the front of this board is a 12×8 grid of red LEDs – that's 96 little blinking lights that you can program however you like. They could be used to show a graph, text output, or other really useful addition to your project; however, in this instance, we will look at how to make them display a simple animation.

The LEDs can be either on or off – there's no inbuilt ability to control brightness on this matrix – so each LED needs 1-bit of storage. The display has 96 LEDs, so it requires 96 bits, which is 12 bytes. The most convenient way of handling 12 bytes in C is three 4-bit integers, which is what the library uses.

You can manually create these three integers if you like. Back in the olden days of programming, it was common to use grid paper and a pencil. Colour in the squares you want, and then you can work out what bits should be set, and from that, calculate the number that represents these bits. However, it's no longer the olden days of programming – we now have tools to help us do this. In fact, Arduino has created one specifically for this LED matrix – you can find it at **ledmatrix-editor.arduino.cc**.



Right
There are probably more practical uses for the Arduino UNO R4 WiFi, but this is a fun way of exploring the LEDs

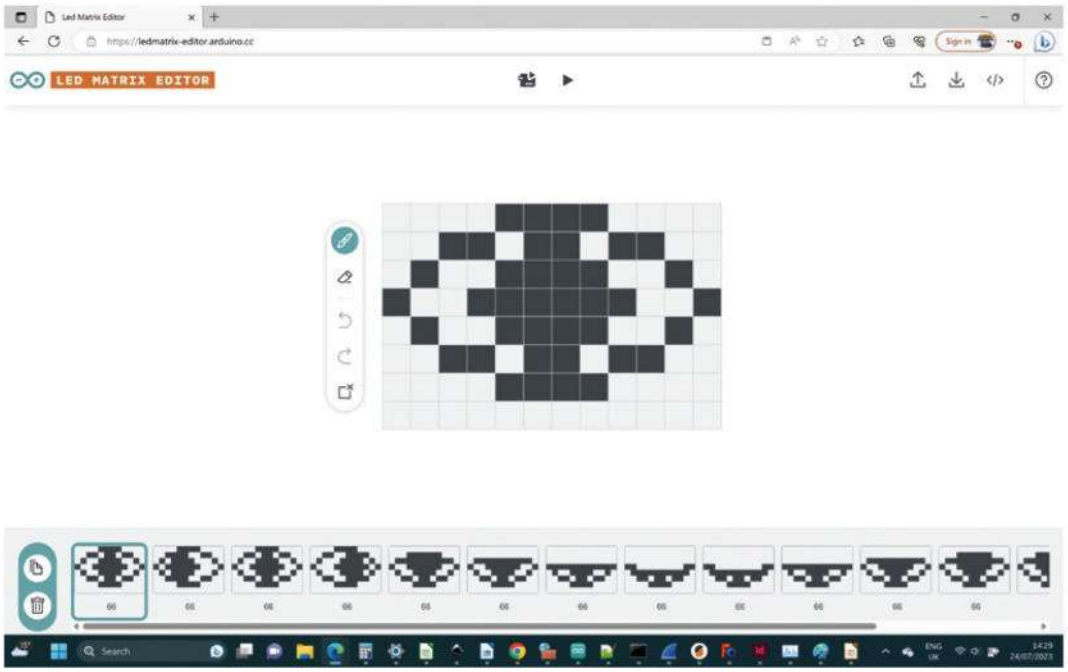
We found there are some good things and bad things about this tool. The good being it's really simple to use. It's set up with the right layout of LEDs, and you can easily set them to be on or off. To create an animation, you just need to add separate frames (shown along the bottom) – these can either be blank or you can copy previous frames. The bad thing about this tool is that it can be a bit flaky. We've had frames disappear, appear in the wrong order, not download properly, and such like. As long as you keep copies of your work as you go along, this isn't too much of a problem.

There are two ways to download your animation. If you click the download icon, it'll download an MPJ file. This is an editable copy of the animation that you can re-upload to the site to make changes to. If you click the code icon, you get a header file with the details for your animation. This is not a cloud tool, so no animations are saved online – you must keep an offline copy or it will be gone when you close the web page.

You may notice some numbers underneath the frames. As far as we can tell, these are the amount of time in milliseconds that the frame should appear on the LEDs. However, this doesn't seem to be very reliable, so we'd recommend leaving these to the default (66) and changing them in the generated code.

You can create whatever design you like. We've generated one for an eye that looks left, right, and blinks. We've uploaded the MPJ file (that you can use if you want to see or edit our design) here: **hsmag.cc/blinkmpj**, and the H file (that you can include in your code) here: **hsmag.cc/blinkh**.

If you look in the H file, you'll see the data, which is a two-dimensional array:



Left
The LED Matrix Editor can also be used to preview your animation either on the website, or by streaming it to the board itself

```
const uint32_t blink[][4] = {
        {
                0xf036c4f,
                0x29f94f23,
                0x6c0f0000,
                66
        },
```

Each entry consists of four unsigned 32-bit integers. The first three contain the details for the LEDs, and the final one contains the time that the frame should be shown for. As previously mentioned, it might be possible to set this in the web interface, but we've had trouble doing this reliably. The easiest option is to adjust them here to whatever you like.

PUT ON A SHOW

Now you've got your animation, we need a sketch to display it on the board.

The basic sketch for this is:

```
#include "Arduino_LED_Matrix.h"
#include YOUR_H_FILE

ArduinoLEDMatrix matrix;


void setup() {
  Serial.begin(115200);
  matrix.begin();
  matrix.loadSequence(YOUR_ANIMATION_NAME);
  matrix.play(true);
}

void loop() {

}
```

The only modification that you need to do here is change **YOUR_H_FILE** and **YOUR_ANIMATION_NAME** to the appropriate things. If you're using ours, that's **blink.h** and **blink**, respectively (the latter one is the name given to the data structure that holds the animation; by default, this will be the same as the file name).

You also need to copy the H file into the directory. You can do this in the Arduino Editor using Edit > Include File.

With this done, you just have to upload the sketch to your Arduino UNO R4 WiFi, and you should have an eye blinking back at you. What you do with your blinking eye is up to you.

FROM THE BEGINNING

In this article, we've assumed you've got your UNO R4 up and running with the Arduino IDE already. If you're not there yet, follow the Getting Started guide from Arduino available at **hsmag.cc/unor4getstarted**. This will guide you through installing the software and making sure it's configured correctly for your boards.

TUTORIAL

Do AIs dream of electric butterflies?

Can computers generate aesthetic flair?

Ben Everard

@ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

While the 3D printer is probably the most iconic tool of modern making, it's the laser cutter that's the workhorse. It's far faster than 3D printing, more consistent than hand tools, and can use a wide range of materials. It's relatively easy to put together a design that can be laser-cut – it depends a bit on your software, but the starting point is usually a vector image, something like an SVG. Software, like Inkscape, makes it easy to create functional designs. However, we've often found it hard to make things that look great.

With the advent of AI tools, a new realm of creative possibilities has opened up. While AI tools cannot yet create fully functional designs, they excel at generating visually intriguing and unique patterns.

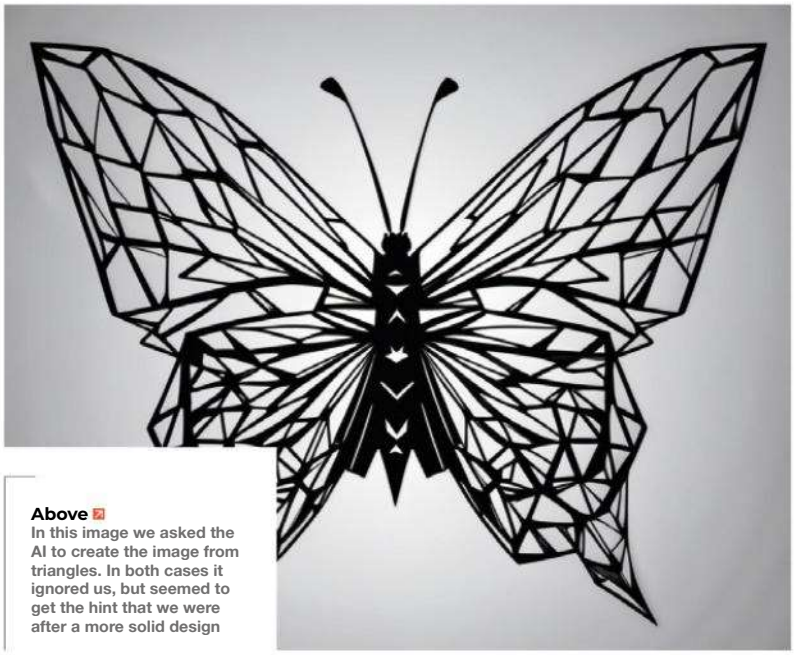
The Stable Diffusion XL AI model is a powerful tool known for its ability to generate stunning images. In this article, we will explore how this powerful combination empowers us to craft laser-cuttable patterns. However, it has one limitation – it can only produce raster images. To transform these images into laser-cuttable patterns, we need vector files. But worry not, for we have a clever solution up our sleeves! By creating bold black-on-white patterns, we can easily trace them using Inkscape's Trace Bitmap option, converting raster images into vector files suitable for laser cutting.

We've decided to use a butterfly as our test-piece. The butterfly's intricate wings provide the perfect canvas for exploring a wide range of shapes and patterns. The first step in creating these wings is finding the right text to prompt the AI into generating a suitable image.

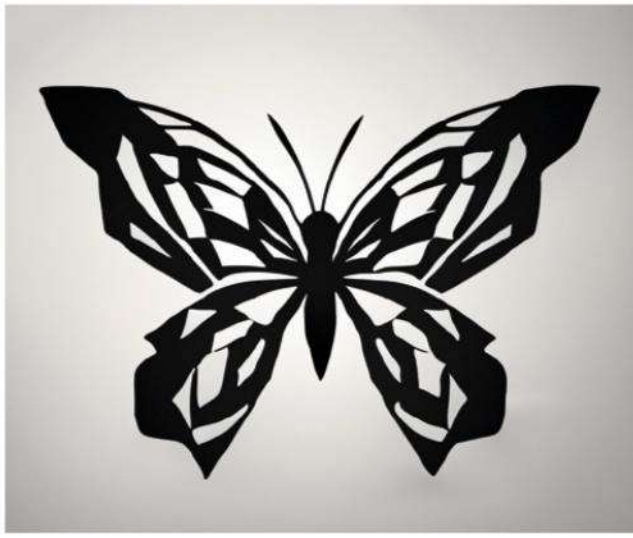
The big problem is that the AI didn't really understand the concept of laser-cuttable tolerances, and structural integrity problems that come with parts

Above
Sometimes the AI seems to forget what a butterfly looks like

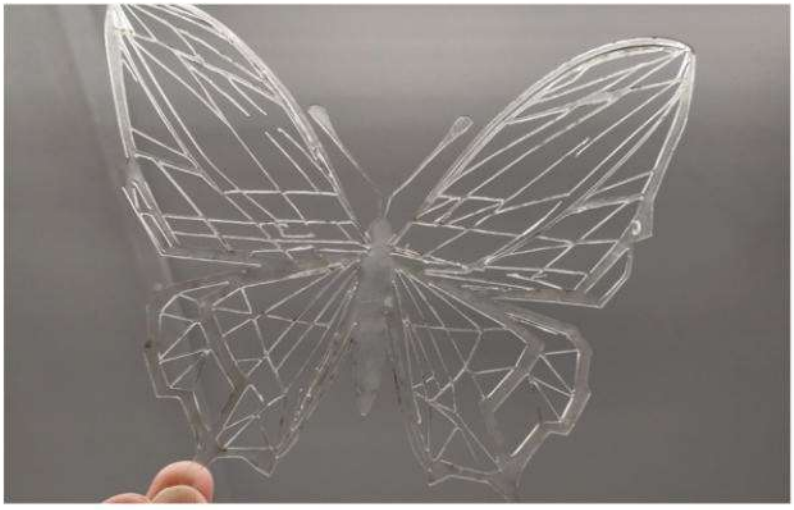
Below
The concept of needing the design to be one contiguous piece is just too complex for the AI to understand, so it sometimes produces pieces like this that look great, but just wouldn't work



Above
In this image we asked the AI to create the image from triangles. In both cases it ignored us, but seemed to get the hint that we were after a more solid design



Above
We asked for the wing pattern to be geometric. We're not sure it followed the brief, but we like the results



Above and left
In clear acrylic, they capture the light and do look great, even if they are a little fragile

that are too narrow. We needed a way to persuade it to use chunkier lines, and it proved tricky to find the right combination of words to enable it to do this.

Our initial experiments, with straightforward prompts like 'Use thick lines', didn't yield suitable images. However, when we provided specific shape instructions, the AI's creative output improved significantly.

For instance, prompts like 'A black silhouette of a majestic butterfly with geometric shapes cut out to create the wing pattern on a white background', and 'A black silhouette of a majestic butterfly with triangle shapes cut out to create the wing pattern on a white background' produced impressive outcomes. The incorporation of geometric elements added both depth and intricacy to the design, making it more visually captivating.

You might notice the word 'majestic' in both of those prompts. We found that we got much more intricate designs when we included it. Omitting this word often led to more icon-like designs.

It's pretty challenging to get an image-generation AI to produce a usable laser cutter design, but it's not impossible. Some of the results we had looked great and cut well. Some looked great, but were too fragile to work well. Some were just outlandish. However, within a short amount of time, we were able to create a design far better than we'd have been able to by hand. An artist specialising in vector art would, no doubt, be able to craft much more interesting images, but that's not really the point. In its current state, AI isn't going to replace artists – it may or may not reach that point in the future – but it does allow people with limited artistic skills to create significantly better work than they would otherwise be able to create.

Water kefir

Make fizzy drinks at home

Ben Everard

@ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.

When you talk about fermenting drinks, people often assume you mean making alcohol, but Saccharomyces cerevisiae, the species of yeast used in bread and beer, isn't the only microscopic organism that is used to ferment drinks.

Water kefir (also known as sugar kefir or tibicos, but not the same as milk kefir) is a ferment that's done with a range of microbes including yeasts and lactic acid bacteria. The result is a tangy, fizzy drink. It's likely to have a little alcohol in it, but not much.

The first thing you'll need is a water kefir culture. This is known as 'grains', but they're more like little blobs of jelly. Inside the grains are a variety of bacteria and yeast. The grains are naturally formed by this colony of symbiotic microbes, and they'll multiply as you make drinks.

You can get your first culture either from someone who already ferments and has some spare, or from an online shop.

This colony needs a source of energy – typically sugar, but any sweet liquid will do. In order to be healthy, they also need minerals. You can use unrefined sugar, or add a bit of molasses for these. Some people also add dried fruit.

> **You can get your first culture either from someone who** already ferments and has some spare, or from an online shop

The basic process is to prepare your sugar-water – we use 130 grams of sugar per litre of water, though you can use more or less. Most of this sugar is consumed during the ferment, so you're not actually drinking much sugar. Only make as much sugar-water as you have grains for – you'll need about two tablespoons per litre. Your grains will grow and multiply as you make more batches, so don't worry if you don't have many at first – you'll soon be inundated with them.

CHLORINE

Most of the water we drink has chlorine in it to kill any bacteria. For most purposes, this is great, but when we're trying to culture bacteria, it's not quite so ideal. The best solution here is to use water without chlorine. However, bottled water can get expensive (and may actually have too many minerals, which can cause problems), and filtered water can remove too many minerals.

We've been using plain, chlorinated tap-water, and have been successfully fermenting kefir. We give it a good glug of lemon juice, as this will help neutralise the chlorine. You can boil it and leave it to cool, or simply leave it to stand in an open jar for a day.

Add your kefir grains and cover loosely. You need to let gases escape, but also want to avoid dust and insects getting in. Depending on how lively the grains are and what the temperature is, you should see it go cloudy and start bubbling in a day or so. Leave it to bubble away for a few days – again, this depends on the temperature, but around two to four days. Taste a little and, ideally, you're waiting until it's a bit sweeter than you want the final drink to be.

Once it's reached this stage, strain it into pressure-safe bottles. When you're getting started, old plastic fizzy drink bottles are easiest because you can test how much gas is in them. At this point, you can add flavours such as blended-up fruits. Pop the tops on and leave to ferment further. Give the bottles a squeeze a couple of times a day to see how the pressure is building. Once they reach about the same hardness as an unopened bottle of fizzy drink, pop them in the fridge. This will slow down the fermenting, but not stop it completely. Once it's cool, drink and enjoy.

After you've bottled one batch, you should have the grains left in your strainer, and these can go straight →

ALCOHOL

Kefir does include yeast and sugar and is fermented in anaerobic conditions. This means that there's likely to be some alcohol in the final product. How much is a very difficult question to answer. There's not an easy way to test the amount of alcohol in a liquid. Home brewers measure the specific gravity of a liquid, which tells you how much sugar there is in it. If you're brewing using only yeast, then you can calculate the amount of alcohol in the drink by seeing how much the proportion of sugar has changed. However, in a drink like kefir, where there's both yeast and bacteria, there's no way of knowing how much of the sugar has become alcohol and how much has become lactic acid and other ferments.

The actual amount of alcohol in a ferment is likely to be very dependent on the exact culture you're using and the environment it's in. It certainly shouldn't be very alcoholic (under 2% or so), and it might be barely alcoholic at all. Unless you're drinking pints of the stuff, or have some other reason to avoid small amounts of alcohol, then the amount of alcohol isn't likely to be relevant.

Above
The grains are squishy colonies of yeast and bacteria

Right
A scattering of dried fruit provide nutrients for the microorganisms

SAFETY

The biggest risk with sugar kefir is pressurised bottles. If you leave it to ferment too long, the gases may build up enough to explode. This is bad if you've used plastic bottles, and really bad if you've used glass bottles. The exact amount of time it takes to reach this point depends entirely on what you're fermenting, what the temperature is, and how lively your culture is. There's no substitute for checking regularly.

Some people have an adverse reaction to kefir. It contains a lot of yeasts and bacteria. While these are beneficial microbes, a sudden influx can upset the balance of intestinal flora in some people. To minimise the chances of this happening, it's best to start small and build up. This is also the best way of building up your culture when you're first starting. If your first few batches are just a litre or so, that's drunk between a couple of people over a couple of days, then that gives your body time to acclimatise.

Water kefir falls into the category of probiotic, and here in UK, the NHS gives the following advice:

"for most people, probiotics appear to be safe. If you want to try them, and you have a healthy immune system, they shouldn't cause any unpleasant side effects … If you have an existing health condition or a weakened immune system, you should talk to a doctor before taking any probiotic supplements."

QUICK TIP

Metal is toxic for your kefir grains, so you should avoid any prolonged contact. It's fine to give it a bit of a stir with a metal spoon, or strain briefly in a metal sieve, but avoid anything more than this. Don't store your grains in a metal pot, or ferment in a metal bowl.

> **As with most things food-related, it all comes down** to personal preference, so experiment

into another batch. If you don't want to start another batch straight away, pop them in a bit of sugar-water and put them in the fridge. They're a living product, so you can't just store them indefinitely – you need to let them keep growing, and keep feeding them.

A couple of variations are as follows:

GINGER BEER: (sort of, see box on the opposite page). Start by boiling ginger root in one third the final quantity of water – we used 50 grams per litre. Once it's boiled for a few minutes, strain out the ginger and add the sugar, and stir it until it's dissolved. Leave to cool for a bit, then add the remaining two thirds of the water. This should cool it down to around warm bath temperature, at which point it's safe to add to the kefir grains.

LOW-ALCOHOL CIDER: This one is super-simple. Just use apple juice. Pour it over your kefir grains,

Above
The grains sit at the bottom of the jar and inoculate the liquid

GINGER BEER

Traditionally, ginger beer is made using ginger beer plant, which is very similar to water kefir. It's also a symbiotic culture of yeasts and bacteria which make up jelly-like lumps in the liquid and, for most purposes, you can use them interchangeably (as we have done here). Neither of these should be confused with ginger bug, which can also be used to make ginger beer and is also a culture of yeasts and bacteria, but ginger bug doesn't form lumps. That said, ginger bug can also be used in a very similar way, and can be developed using just a root of ginger and some sugar-water. If you don't have access to water kefir, this could be an option.

then ferment as normal. The end result is like a light cider, and is particularly delicious on a summer evening.

As with most things food-related, it all comes down to personal preference, so experiment to find what you like. You can enjoy delicious fizzy drinks all year round without needing any complicated equipment beyond a few bottles, some sugar, and some strange, squishy grains.

BENEFITS

Water kefir makes a delicious drink, and that is our main reason for making it. However, it may have other benefits.

If you read about water kefir (or indeed any natural ferment) online, then you'll be inundated with health claims, usually presented without evidence.

Let's start with what we do know. A healthy gut microbiome (the range of microbes living in your intestines) can have a significant impact on your health in ways you might not expect – including on your immune system and mental health.

We also know that water kefir contains a range of microbes that can also thrive in your gut. Beyond this, it's hard to say anything definitive. Kefir (or other fermented products) might help diversify or increase your microbiome, particularly if for any reason it's been damaged (such as through illness or medications). It's also possible that your microbiome already contains a wide range of bacteria which leave the kefir bacteria few places to thrive. While it might help with your gut microbiome, water kefir certainly isn't an instant fix for any and all intestinal complaints.

TUTORIAL

Convert an old flash-gun into a web-controlled light

Upcycle old photography tech with a Raspberry Pi Pico and CircuitPython to make a browser-controlled light that brings colour and depth to your photos

Rob Miles

@robmiles

Rob Miles has been playing with hardware and software since almost before there was hardware and software. You can find out more about his so-called life at **robmiles.com**.

The author picked up the flash-gun in Figure 1 at a market stall. Photographers use flash-guns to brighten their pictures. The flash clips onto the top of the camera and provides a burst of light when a photo is taken. A flash like this would have been quite expensive back in the 1970s when it was made. Modern digital sensors are much more sensitive to light than photographic film, so we don't need the powerful flash-guns of the past quite so much. These days, you are more likely to use a portable light to add colour to a scene, which is what the author plans to use the light for. It will look exactly the same as the original, but it will not flash. When it is switched on, it will connect to Wi-Fi and host a website on the local network that lets you control colour and brightness.

Figure 1 The flash-gun is from an age when they used to put component names on the outside of the device. A thyristor was quite a big thing at the time the flash-gun was made

HOSTING A WEBSITE IN A LIGHT

Figure 2 shows the web page that is hosted by Pico W fitted inside the light. You can set the colour of the light by using the sliders to adjust red, green, and blue intensity or hue, saturation, and brightness. The master brightness slider controls the overall brightness of the light. The Pico W runs an HTTP server which serves out the web page. The latter contains JavaScript code to read the slider values and send web requests back to the Pico W. The web requests contain the required colour and brightness settings. Let's look at how this works. The overall sequence of operation is as follows:

1. The light is switched on. A CircuitPython program starts running in the Pico W. The program connects to the local Wi-Fi, and then starts a web server hosting the web page you can see in **Figure 2**.

2. The user opens the browser program on their device and navigates to the page at the address **http://flashlight.local**. The browser sends a request to the local network asking if anything on the network is called 'flashlight.local'. CircuitPython code in the Pico W responds with the network address of the flashlight and the browser requests the index page for the site from that network address.

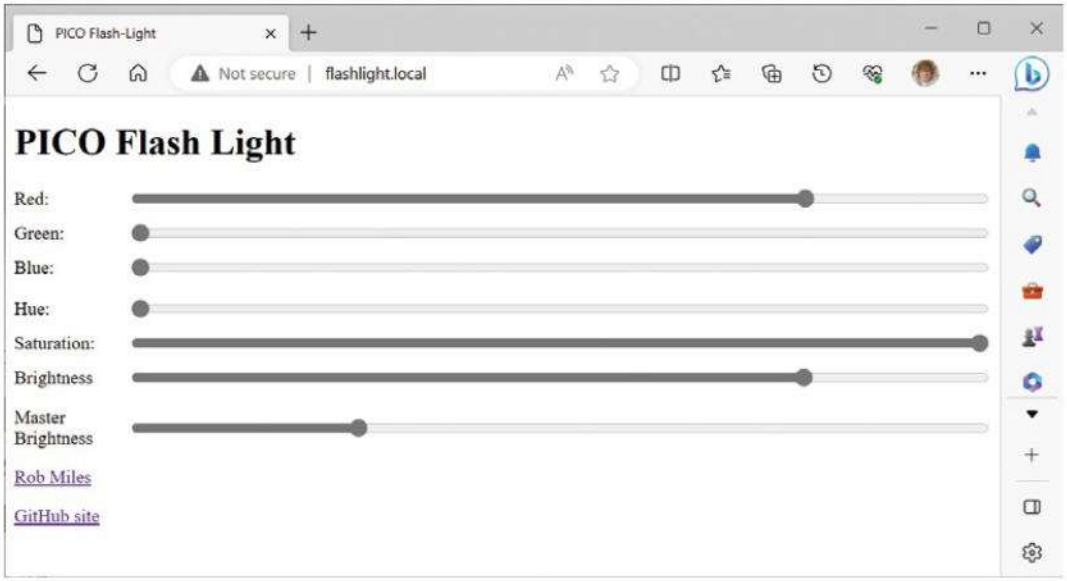


Figure 2
The sliders in the web page fit the width of the browser being used. It works well on mobile phones in landscape mode

3. The web server running on the Pico W returns the contents of the **index.html** file (also held on the Pico W).

4. The browser receives the index page from the light and starts running the JavaScript code in the page. This starts a task that runs five times a second inside the browser. Each time the task runs, it checks the red, green, blue, and brightness values that the user has selected to see if they have changed.

5. If the light values have changed, the JavaScript code in the browser sends a web request to the address **http://flashlight.local/lights**. This request contains the light values which are used to control the light output.

6. The server in the flashlight receives the request and updates the light settings.

7. Further changes to the light settings will result in more web requests, which will update the light again.

SERVING A WEB PAGE FROM A PICO W

The Pico W runs a CircuitPython program to host the light website and respond to incoming messages to control the lights. Note that the code samples in these sections just show you how the code itself works. You will need to include some CircuitPython libraries to get the code to run. You can find all the source code and instructions in the GitHub repository for this project at **hsmag.cc/PicoFlash**.

The first step in hosting a web page on a Pico W is to connect the Pico W to the local Wi-Fi. The code below uses values from the **settings.toml** file stored on the Pico W. When you make your light, you will have to create this file with your network SSID and password values in it.

```
wifi.radio.connect(os.getenv('CIRCUITPY_WIFI_
SSID'), os.getenv('CIRCUITPY_WIFI_PASSWORD'))

print("Connected to WiFi")
```

Now that the Pico W is connected to the network, it can start an mDNS server and configure it to respond to requests for 'flashlight.local'. The name mDNS stands for 'multicast Domain Name System'. It is a way for devices to find each other on a local network. When the browser sees an address that ends in '.local' (for example, **http://flashlight.local**) it sends an mDNS broadcast onto the local network asking 'Anyone here called flashlight.local?' The flashlight can respond to this request with a message containing their network address. The browser can then connect directly to that device. The code below sets this up. If we want to use multiple lights on our network, we just have to give them different hostname values.

```
mdns_server = mdns.Server(wifi.radio)
mdns_server.hostname = "flashlight"
```

Now we can create our web server. The web server uses a network socket to receive and transmit messages. The statement below creates that socket:

```
pool = socketpool.SocketPool(wifi.radio)
```

Now we have the socket, we can create a web server that will connect using it. The code below does →

YOU'LL NEED

- **An old flash-gun** (or you can put the light into anything you fancy)
- **Raspberry Pi Pico W**
- **A bright light** You can use a super-bright NeoPixel or a Pixie 3W LED (**adafruit.com/product/2741**). The author used the Pixie as this has overheat detection
- **A level converter chip** (search for 'Arduino level shifter' on your favourite e-commerce site)

this. The `Server` constructor is supplied with three parameters: the socket to use, the name of the folder which contains the site files, and a debug option. The statement below creates the server and tells it that the site folder contains all the HTML files. It also turns debug on so that we can see requests on the Python console.

```
server = Server(pool, "/site", debug=True)
```

The next thing we need to do is set up some routes. A route is a particular address that a server responds to. The light server will host two routes: the root route (the author apologises) for the index page which is '/' and the lights route which is '/lights'.

> The second route that the Pico needs to support is **the route used to set the colour of the light**

ROUTING FOR THE INDEX PAGE

The index page route returns the HTML for the index page that describes the sliders on the web page, and contains the JavaScript code that makes the page respond to the sliders and send the colour values to the Pico. The code below implements this route. It uses the slightly confusing Python 'decorator' syntax. This can be hard to understand, but the only thing you really need to know is that when the browser asks for the route '/' (the index page of the site), the `base` function will be called with the web request object as a parameter. The index route serves out the index file, so that is what the code does. A `FileResponse` object fetches a file from the site folder which is delivered back to the browser. The **index.html** file is held on the Pico in the **site** folder.

```
@server.route("/")
def base(request: Request):
    return FileResponse(request, "index.html", "/
site")
```

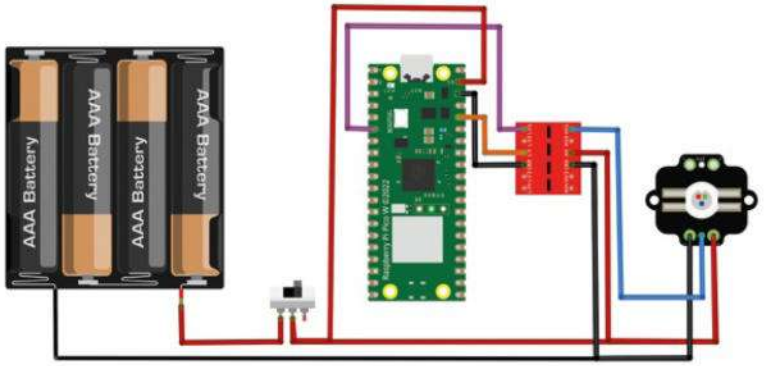


Figure 3
Four 1.5V alkaline batteries would produce 6 volts, which is slightly more than the Pico W is specified to accept on its power input. However, the circuit seems to work OK, particularly if you use rechargeable batteries which only produce around 1.25 volts each

SUPER-BRIGHT LIGHTS

The first version of the light used NeoPixels, which worked OK, but the author wanted a bit more power, so the finished version actually uses Pixie lights (**hsmag.cc/ChainablePixel**).

These can provide up to 3 watts of light output, compared to the 0.2 watts of a NeoPixel. They have other advantages too – they will shut down if they overheat and will automatically turn off if they lose connection with their controller. A single Pixie unit was mounted behind the flash reflector in place of the original xenon flash tube.

ROUTING FOR THE LIGHT COLOUR

The second route that the Pico needs to support is the route used to set the colour of the light. The browser will use this route when it detects that the user wants to change the light colour. When a browser asks a server for a web page, it uses a message called a 'GET' request. The GET request can include 'query' values which are sent as name-value pairs.

The code below implements the 'lights' route in the server and extracts colour values which are used to set the colour of the light. The code then sends a simple response (the word 'lights') to the browser.

```
@server.route("/lights")
def base(request: Request):
    red = request.query_params.get("red") or 0
    green = request.query_params.get("green") or 0
    blue = request.query_params.get("blue") or 0
    master = request.query_params.get("master") or
10
    setColour(red,green,blue,master)
    return Response(request, "lights", content_
type='text/html')
```

Now that the server has been set up, the CircuitPython code running in the Pico can start it running. The code below does this. If the server doesn't start, the code waits five seconds and resets the device to try again.

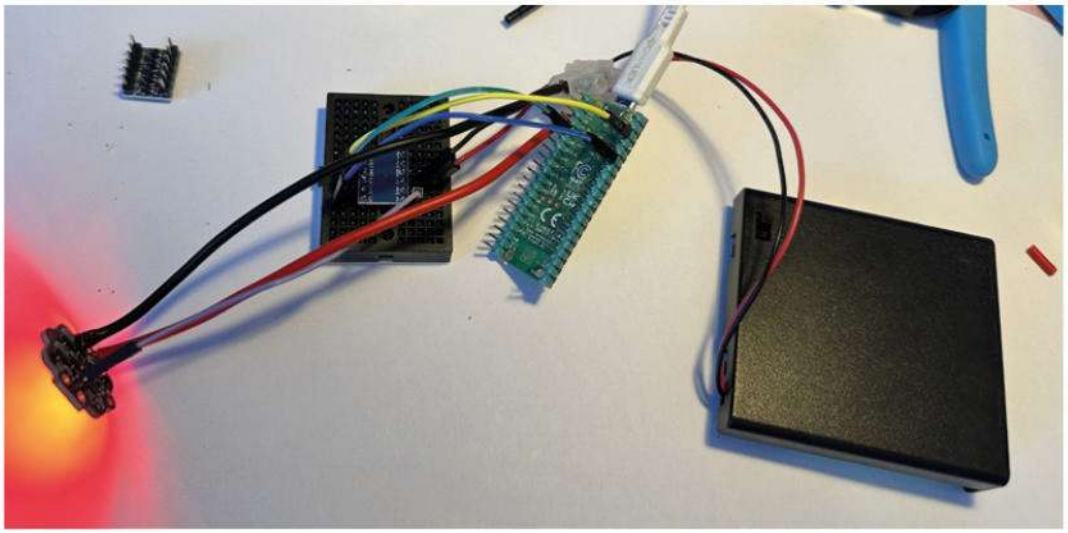


Figure 4
The battery pack in the picture contains four AA cells, matching the power source in the flash-gun and allowing it to be tested with the correct power supply voltage

```
try:
    server.start(str(wifi.radio.ipv4_address))
    print("Listening on http://%s:80" % wifi.radio.
ipv4_address)
#  if the server fails to begin, restart the pico
w
except OSError:
    time.sleep(5)
    print("restarting..")
    microcontroller.reset()
```

The server needs to be polled at regular intervals. Each time the server poll method is called, it will check for any incoming web requests and respond to them. The code below does this. It also calls a function called **colourTick**. The Pixie LED needs to be updated every two seconds or it will automatically turn off the light. The **colourTick** function does this.

```
while True:
    try:
        server.poll()
    except Exception as e:
        print(e)
        continue
    colourTick()
```

WEB PAGE LIGHT CONTROL

The web page in **Figure 2** shows how the colour of the lights is controlled. We now know how this page is served by the Pico in the light. Next, we need to discover how the code in the web page works. This code runs inside the browser and is written in JavaScript (the language of the web). When the browser loads the web page from the Pico, it calls a function called **startReader** to start the value reader. The first thing the function does is set up the address of the server so it knows where to send web requests to the server to set the light colour. To do this, it needs to know the host name of the light. This will usually be 'flashlight.local', but we might change this if we have multiple lights. The code below gets the hostname from the browser, and then creates the address for the lights URL.

```
function startReader() {

    // get the name of the web host
    var hostName = window.location.hostname;
    // make the host address from the host name
    hostAddress = "http://"+hostName+"/";
    // make the address of the lights setting route
    setLightsURL = hostAddress + "lights";
    console.log("Host address: " + hostAddress);
```

Now that the host has been set up, the next part of **startReader** configures the sliders on the page.

Figure 5
The level converter provides four level converted signal paths, but the light only needs one

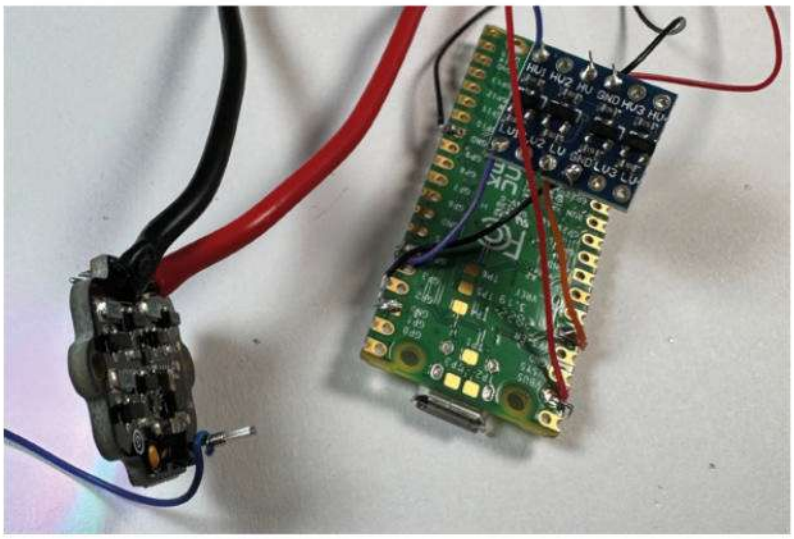


Figure 6
The grey cylinder is the capacitor that is charged up to provide a high-voltage pulse to excite the gas in the flash tube, causing a flash of light when the flash is fired

Above right
Reassembly can be a little tricky as you need to get the battery connectors to stay in place

The page contains sliders that are used to set the intensities of the red, green, and blue components of the light colour. There are also sliders for hue, saturation, and brightness. The author had quite a bit of fun making the user interface automatically update when any of the sliders are changed. To do this, each of the sliders is attached to a function. This is performed in the **startReader** function. The code below shows how this is achieved for the red slider. The first statement sets a variable called **redSlider** to refer to the red slider on the page. The second statement causes a function called **colChange** to run when the red slider is moved by the user. The **colChange** function updates the slider positions on the page. There are versions of this code for each of the sliders.

```
    redSlider = document.getElementById("red");
    redSlider.oninput = colChange;
```

You might think that it would be a good idea to send a web request to select a new colour each time a slider is changed, but this is not the case. A moving slider generates an **oninput** message each time it is changed, so dragging a slider would produce lots of colour change web requests which would overwhelm our little Pico-powered server. A better way to do this is to check the colour values at regular intervals and send a message when the colour values have changed. The statement below, which is the final one in the **startReader** function, tells the browser to call a function called **tick** every 200 milliseconds.

```
    setInterval(tick, 200);
```

The **tick** function checks to see if the colours have changed and sends off a web request to the Pico server if they have. The first thing it does is get the red, blue, green, and master values from the sliders. Then it compares these with the old values. If these have changed, the code creates a web request URL for the 'lights' route which contains the colour values to be sent to the Pico. The **getFromServer** function is used to fetch a page from this URL. The values in the route are picked up by the server running on the Pico as described in the section 'Routing for the light colour' (previous page). You can use this mechanism to send any value from a web page into a Pico.

```
function tick() {

    // get the values
    const red = redSlider.value;
    const blue = blueSlider.value;
    const green = greenSlider.value;
    const master = masterSlider.value;

    // have any changed?
    if (red == oldRed && blue == oldBlue && green
== oldGreen
        && master == oldMaster) {
        return;
    }

    // Update the old values
    oldRed = red;
    oldGreen = green;
    oldBlue = blue;
```

```
    oldMaster = master;

    // build the url
    let url = setLightsURL +
      `?red=${red}&blue=${blue}&green=${green}
&master=${master}`;

     // get from the server
     getFromServer(url, result => {
           console.log(`Got response:${result}`);
    });
}
```

Now that we know how the software works, we can build some hardware.

PICO CIRCUIT

Figure 3 shows the circuit for the light. The Pixie light unit is controlled by serial data sent from GP4 on the Pico. The GP4 port is the output from the serial port on the Pico. The output is sent through a level converter which lifts the 3.3V signal from the Pico to the 5V level accepted by the Pixie. With the circuit designed, it was then prototyped (**Figure 4**) and the control software tested on it to make sure it worked.

DEAD COCKROACH CONSTRUCTION

There is only just enough room to fit the Pico W inside the flash-gun case. The circuitry is constructed in 'dead cockroach' style. This is where the finished circuit looks like a cockroach lying on the bench with its legs in the air. In **Figure 5**, you can see that there are two 'cockroaches': the Pico W and the level converter chip. These were taped together and then slotted inside the flash-gun.

THYRISTOR POWERED

The thyristor in the flash-gun is the three-legged component labelled 'S6095' on the central circuit board in **Figure 6**. Thyristors represented a huge step forward when they were introduced in the 1970s. They allowed transistor-based circuits to switch high voltages and currents. The flash-gun has a light detector on the front which reacts to light reflected from the scene being photographed. When the detector has seen enough light, it tells the thyristor to turn off the flash. This prevents overexposure and saves energy, making the flash-gun recharge more quickly for the next shot. Thyristors and similar devices are now used throughout power switching. The author has fond memories of using a thyristor-powered 'sound to light' unit for many a disco back in the day.

MAKING THE HARDWARE

The first stage of the conversion was to dismantle the old flash-gun. It came apart quite easily; most of the case was held together using metal clips. All the existing electronics were removed and the circuit board holding the power switch was cleared of components so that the switch could still be used but there would be room for the Pico above it. **Figure 6** shows all the different parts. The Pixie was mounted on the back of the flash-gun reflector, and power and data cables run into the main body of the flash where the Pico was mounted just underneath the battery. Be aware that the capacitor could hold a high voltage if the flash was used recently, so don't proceed if you aren't familiar with how to handle this safely.

The hardest bit was getting the battery connectors to stay in place while the two halves of the flash-gun body were put together. It turned out that the best way to do this was to put the batteries into that half of the case, holding the connectors in position. The finished system works well. It can produce very bright lights in a range of colours. It is turned off by a switch on the rear of the flash-gun. You can use this technique to add remote-controlled features to any device large enough to contain a Pico W.

QUICK TIP

If you throw away any parts of a device you are upcycling, you must make sure that they are disposed of properly as electronic waste.

Below
The flash-gun has a Fresnel lens on the front which can be moved forwards and backwards on the flash-gun to allow the light output to be focused at different distances

The sublime art of sublimation

Get started with sublimation printing

Dr Andrew Lewis

Dr Andrew Lewis is a specialist fabricator and maker, and is the owner of the Andrew Lewis Workshop.

The ability to apply repeatable designs to a surface that isn't paper can have a massive effect on your making and prototyping process. Glass, metal, fabric, wood, and other materials can all be printed onto if you have the right bits of equipment. In this article, you'll learn the basic tools and techniques needed for sublimation printing – a transfer printing technique that will let you produce vibrant and resilient designs on a huge range of materials.

Sublimation printing is a multistep process. You begin by creating and printing your design using special ink and special paper. Normally, you will reverse the image when you print it out, because you are going to transfer the image from one surface to another; the front of the image, as printed, will then become the reverse of the image once it's transferred onto the final surface.

The selling point of sublimation printing is that you can print your designs onto shapes and materials that won't fit through a desktop printer. Unfortunately, you can't print straight onto every surface. At the basic level, sublimation printing is about melting coloured plastics to the point where they will bond with other plastics. You transfer an image from the transfer paper onto the final surface using a heated press. The press applies an even, static pressure for as long as the transfer requires. Different materials need different amounts of time to transfer.

Preparing a surface for sublimation printing generally involves applying some sort of plastic coating to a surface so that sublimation ink will stick to it when it gets hot. You can do this preparation yourself but, for common items, it's cheaper and easier to buy premade items called sublimation blanks. There are companies that specialise in →

Left
Sublimation paper has a right side and a wrong side, which isn't always obvious. Some paper is coloured on one side to indicate the back, while other brands just rely on the fact you'll be removing it from the pack in a particular direction. Note that, because this is a transfer print, the design is mirrored so that the text will read the right way around when it's transferred

> **It's cheaper and easier to buy premade** items called sublimation blanks

PICKING A PRINTER

You will need a dedicated printer with specialist inks and paper for sublimation printing. It might seem like a daunting investment at first, but you don't have to spend thousands of pounds to get started.

There are suitable inks available for Epson EcoTank printers like the ET-1810, and you can pick one up online for around £200. Don't be tempted to buy second-hand, as the printer needs to be loaded with special sublimation ink, and completely flushing out any existing ink is not an easy task.

It's well worth paying the extra few pounds for a new machine. Some sublimation suppliers like Ink Experts (**hsmag.cc/InkExperts**) will offer bundle deals with a printer, ink, and paper all together for a very reasonable price.

Most suppliers offer both hard surface papers (for ceramics, wood, and acrylic sheet) and soft surface papers (for fabrics) for sublimation printing.

You can technically use plain paper for sublimation printing, but the results won't be very vibrant and the paper will likely stick to the sublimation blank. It's much better to use dedicated sublimation papers for the material you're going to be sublimating onto.

UNDER PRESSURE

A multipurpose press can be set up to accommodate many different types of sublimation blank. The photo below shows the same press configured as a flat press for fabric, plastic, or metal sign blanks. It can also be set up as a hat press, with a curved plate specifically designed to match the curve of a trucker's hat. The lever handle at the front-top allows you to lock the plate into place, while the hand-wheel on the top allows you to set the pressure. The control box at the side lets you configure the temperature and set a countdown timer.

A heated press is essential for sublimation printing. Good-quality sublimation prints need you to apply a steady, even pressure for a set amount of time. This isn't something you can do properly with a domestic iron. The key to even pressure is to use a press that's designed for the object you're pressing your design onto. Cups and tumblers need a cylindrical press, hats and bags need a shaped press, shirts and signs need a flatbed press, and dinner plates need small, round plates that can apply pressure just to the centre of the plate to avoid breaking the rim. While there are dedicated presses for each of these items, you can also purchase an entry-level sublimation press with interchangeable plates for different items. For a hobbyist, these small presses are ideal.

Left
A multi-purpose press can be configured for many types of sublimation blank – here it's set up as a flat press

QUICK TIP

Sublimation inks work best when printed on a white background. If you're printing on coloured materials, you might want to prime the area with a white undercoat or heat transfer vinyl.

QUICK TIP

For the best results, preheat your press before you start printing by closing the hot press for a minute to make sure the top and bottom plates are both heated properly.

Right
The edge of this design is faded because the heating element wasn't able to apply enough heat and pressure near to the handle. This isn't an issue if you are using a sublimation oven with a silicone grip

> **The press applies an even, static pressure for as long** as the transfer requires

FEEL THE HEAT

As you advance with sublimation, you might consider getting a dedicated sublimation oven. A sublimation oven will allow you to process multiple items at once, with even a small oven accommodating six mugs or tumblers at the same time. The oven also allows you to make more advanced designs on mugs that aren't possible using a press. The design of a mug press means that it's very difficult to apply pressure to the area nearest the handle of the mug. Mugs made on these presses typically have a noticeable blank area on the side of the mug with a handle. However, a mug prepared with a silicone clamp around it for a sublimation oven can be printed all the way around, with the design continuing underneath the handle (with a little bit of practice). It's important to note that you cannot simply use a domestic oven for sublimation. The sublimation process releases gasses that could taint any food prepared in the oven, so as tempting as it might be, don't even think about warming up your sausage roll in the sublimation oven.

QUICK TIP

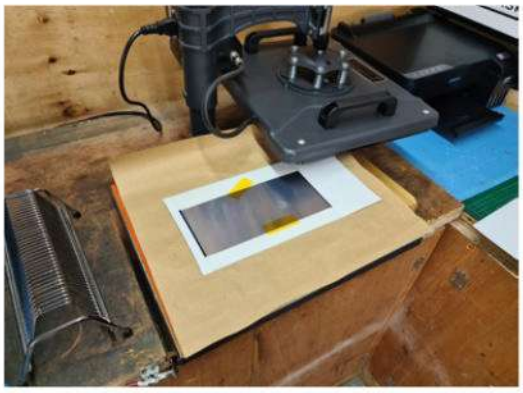
A good-quality air fryer will work just as well as an industrial sublimation oven.

EXTRA EQUIPMENT

In addition to the printer and heated press, you'll need a few other items: some heat-resistant foam to pad your press (this is often included with the press), Kapton tape to hold your designs in place on sublimation blanks, and some silicone baking sheets. The baking sheets help keep your press plates clean, since the sublimation process can cause some ink to leak around the edge of or even through the sublimation paper. If this happens, it's easier to cut another sheet of baking paper than it is to try to clean molten ink off a hot press. Finally, get some heat-proof gloves (silicone baking gloves are ideal) so that you can pick up hot blanks and move them around without burning your fingers.

producing blank stock for making signs, cups, key rings, coasters, cutting-boards, and a whole variety of other items. If you're just starting out, these items are the best choice to practise on.

This article covers the basics of sublimation printing on a number of different materials, but when it comes to sublimation printing, this is only the beginning. In the next article, you'll see how to sublimate onto difficult and uncoated materials like glass sheets, prepare raw materials to create your own sublimation blanks, and protect your prints to make them even more resilient than they already are.



Above
It can feel counter-intuitive to place the sublimation side of the blank away from the heating element, but it can produce much better results. Here, you can see the printable on the bed of the machine, ready to be pressed, with the sublimation paper folded and taped into place. The excess paper on the end of the blank makes it easier to pick up the hot sheet. The paper can be removed easily once the sheet has cooled

A SOFT SIDE

Soft materials are more challenging for the amateur sublimation printer to make. A soft, stretchy expanse of fabric distorts easily, and it's more difficult to make sure that your sublimation paper stays in the right place. The pressing process is much less forgiving, because any slight change in pressure or the position of the paper will create a blurry print or a double image. Use heat-proof tape to fix the sublimation paper in position, and do not even touch the press until the sublimation process is complete. Your choice of sublimation paper and blank will have an effect on the sublimation settings, but 160°C for 50 seconds is a good place to start if your blank doesn't come with any other instructions.

PREHEAT

The biggest issues with ceramic sublimation blanks is that they can make it difficult to heat your object evenly, and the blanks will crack under excess pressure. Press ceramics at a lower pressure, and preheat them in the press before you apply the sublimation paper. For a mug, the extra mass at the base of the mug will soak up more heat than the thin sides, so make sure the base of the mug is hot. If you don't have any instructions specific to your blank, preheating in the press for two minutes, then pressing at 180°C for four minutes at temperature (so start timing once the press is at the target temperature) with a medium pressure should give you a reasonable result in most cases. Different mug styles and presses will affect the time, so some experimentation will be needed.

HEAVY METAL

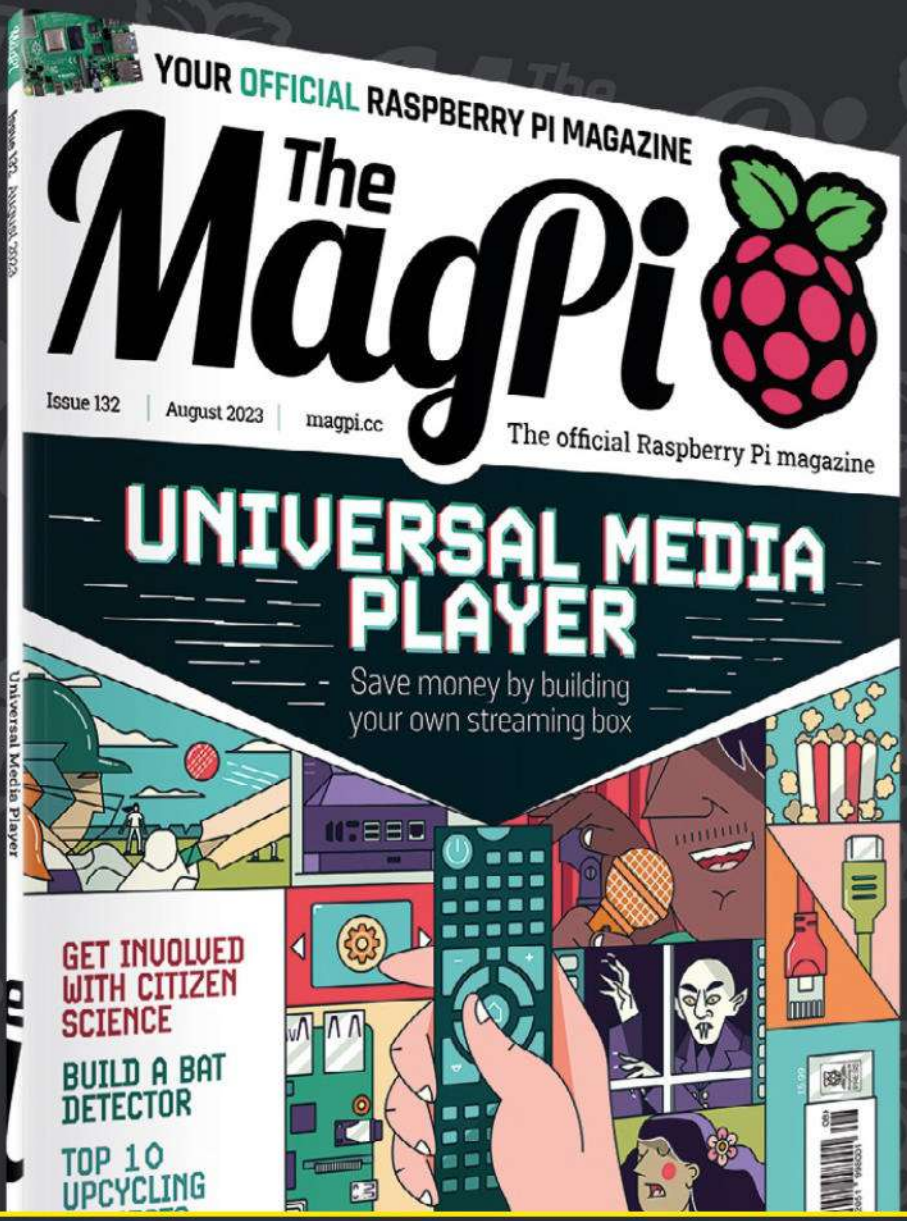
Metal sublimation blanks can be single- or double-sided, and typically the sublimation surface will have a protective film covering it that you need to remove before you apply the sublimation paper. It's easiest to fold your sublimation paper around the blank and fix it in place with Kapton tape. The folds around the edge of the blank will help hold the paper in the right position and an overhanging piece of paper at the end will make it easy to remove from the press when it's hot. For single-sided blanks, place the metal sheet into the press so that the side with the sublimation coating is facing away from the heating element. This means that the entire sign will need to heat up before the sublimation process occurs. When you remove the sign from the press, put it to one side and let it cool before removing the sublimation paper. Suggested settings for metal stock are 180°C for 70 seconds with a medium to high pressure.

Above
With care, you can get vibrant colours on many different surfaces

QUICK TIP

Cleanliness is important with sublimation printing. You don't want any grease, grit, fluff, hair, or other rubbish contaminating your sublimation paper or blanks.

DON'T MISS THE **BRAND NEW** ISSUE!



SUBSCRIBE FOR JUST £10!

- **THREE!** issues of The MagPi
- **FREE!** Raspberry Pi Pico W
- **FREE!** delivery to your door

+ FREE RASPBERRY PI PICO W*

Three issues and free Pico W for £10 is a UK-only offer. Free Pico W is included with a 12-month subscription in USA, Europe and Rest of World. Not included with renewals. Offer subject to change or withdrawal at any time.

magpi.cc/subscribe

HackSpace
TECHNOLOGY IN YOUR HANDS

FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

ONLY THE BEST

Kits for young or new electronic hobbyists

A collection of kits that are perfect for jumping into DIY electronics

By **Marc de Vinck** @devinck

Do you know someone who is ready to get started in the world of DIY electronics? Are they young? And are you unsure of where to even start? Well, that's where this Best of Breed comes into play. I've gathered a series of kits that make for a great progression of learning in electronics. This certainly is not the only path to learning, and some may be too advanced or too simple for your needs, but there is something for everyone, even advanced builders.

You will need some simple tools for a few of the more advanced kits. Just don't be intimidated by having to select the exact right tools like a soldering iron, solder, and wire cutters. There are plenty of beginner soldering kits out there, and lots of information online. And realistically, the first tools you buy aren't going to be the only ones you use in the future. Just don't buy junk. But you also don't need the best and most expensive tools either. The basic tools you purchase when getting started quite often will become your 'backup tools' as you learn exactly what you want, and expand your collection. The most important part when starting out is to make sure to be safe. Always wear eye protection, and work in a well-ventilated area when soldering. Now, let's get started!

Business Beasts - LED Craft Kit vs Bearables Fox Kit

PIMORONI $5.40 | pimoroni.com

PIMORONI $17.46 | pimoroni.com

This electronics kit proves that you can never be too young to start learning about electricity. Sure, you might need some adult supervision, but there is no need for a soldering iron or other tools. The Business Beasts LED Craft Kit is perfect for kids three years and older. OK, so maybe you can be too young, like one or two. But still, an electronics kit that, with a little adult supervision, allows kids to explore electricity and light up a colourful LED is fun!

The kit includes copper tape, five LEDs, an emoticon binder clip, and four different character cards, and all for a very affordable price. So, grab a battery, your favourite Business Beasts card, check out the simple instructions, and get to building your first simple circuit that lights up an LED.

Here's another fun kit for kids eight years of age and older. The Bearables Fox Kit allows you to build a fun, colourful LED badge in the shape of a fox. The kit includes a fox badge with twelve multicoloured LEDs already soldered on, a light sensor in the shape of a flower, 3m of stainless steel conductive thread, three sewing needles, a CR2032 3V coin cell battery, two sticker sheets, and instructions all housed in a reusable box.

You'll learn about electricity and conductivity while sewing the circuit, which allows you to control the brightness of the fox badge based on the flower sensor's input. When it gets dark, your badge will turn on and display a beautiful pattern of light. You can even select one of several different patterns that have been pre-programmed on the PIC16F1503 microcontroller. No programming or soldering is needed, but there are some sharp needles, so adult supervision is required.

VERDICT

Business Beasts - LED Craft Kit

Doesn't get much easier than this kit.

9/10

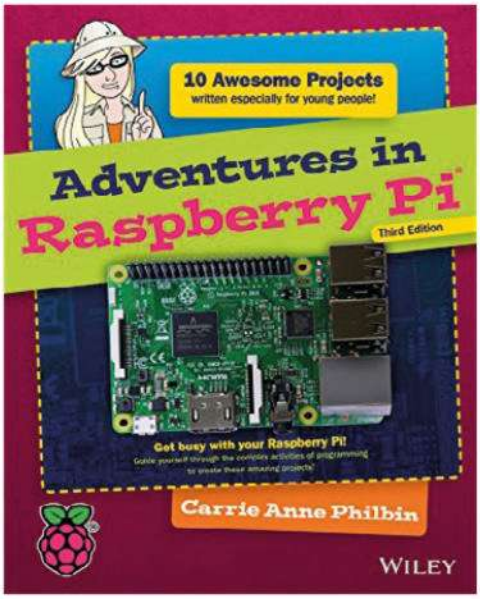
Bearables Fox Kit

A great next step.

10/10

Adventures in Raspberry Pi, 3rd Edition

WILEY $24.12 pimoroni.com



***Adventures in Raspberry Pi* is a great way to get started learning about how to program the ever-popular Raspberry Pi. And you might think, 'Sure, but can you get a Raspberry Pi?'** And the answer is Yes! Pimoroni, at the time of this writing, has plenty of Raspberry Pi 4 Model Bs in stock.

The book, written by Carrie Anne Philbin, allows anyone, even those with no programming experience, to get up and running quickly. You'll learn the basics of the Raspberry Pi, and move on to program games, code up some music, and even build a jukebox. The book was designed specifically for kids aged 11 to 15, but I bet both younger and older kids could learn a few things, too.

VERDICT

Adventures in Raspberry Pi, 3rd Edition

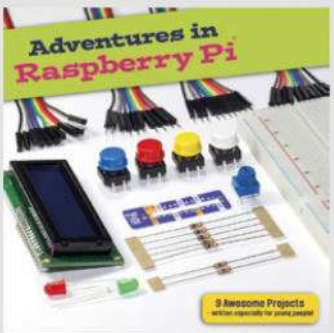
When it's time to introduce the Raspberry Pi.

9/10

ADVENTURES IN RASPBERRY PI – PARTS KIT

PIMORONI $19.81 pimoroni.com

Did you pick up a copy of *Adventures in Raspberry Pi* and would like a simple way to gather all the components needed to complete the projects? Don't worry, Pimoroni has you covered with the Adventures in Raspberry Pi – Parts Kit. Just like it sounds, this kit contains all the parts needed to build the projects in the book. And while you are at it, pick up a Raspberry Pi while they are in stock, too!



Gemma M0 Starter Pack

ADAFRUIT | $27.95 | adafruit.com

After you build your first sewable kit, it's time to introduce a slightly more complex kit with the Gemma M0 Starter Pack by Adafruit. This kit allows you to create your own customisable LED wearable light show thanks to the included Gemma M0 wearable electronic platform, four Flora RGB NeoPixels, a coin cell battery holder, stainless thin conductive thread, needles, small alligator clip test leads, and a USB cable. Yes, that's a lot of parts! Awesome! You just need to add a coin cell battery and you're ready to go.

Once you decide what you would like to create, you can then move on to programming the Gemma M0. It's easy to program in a variety of different environments, including the Arduino IDE, CircuitPython, or even MakeCode. Adafruit has great tutorials to get you started. So, when you are ready to up your game and start programming electronics, take a look at this kit – it's a great starting point!

VERDICT

Gemma M0 Starter Pack

When you are ready to add a little code to the mix.

10/10

Drawdio kit - v1.1

ADAFRUIT | $17.50 | adafruit.com

One of the first kits I built with my kids was the Drawdio by Jay Silver and manufactured by Adafruit. I even made a YouTube video over 14 years ago about the build with my kids that garnered 80,000 views or so. I miss those days!

OK, back to the kit! The Drawdio uses a 555 timer to create a single tone. Couple that 555 with some clever circuitry, and the tone can be varied by the resistance introduced by the graphite in your drawing. It's magical! My kids loved it, and you can even hack it to make other musical instruments. Head over to the Adafruit site to learn more and see just how fun Drawdio can be.

VERDICT

Drawdio kit - v1.1

You can't go wrong with this kit. So much fun!

10/10

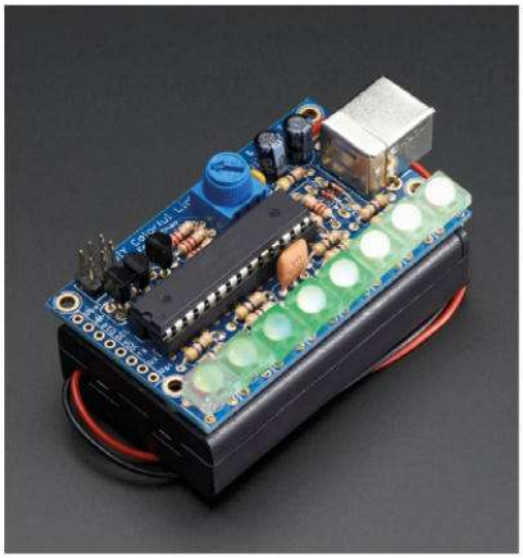
BEST OF BREED

MiniPOV 4 Kit

ADAFRUIT | $17.50 | adafruit.com

Once you have honed your soldering skills with the Drawdio kit, it's time to make another fun kit! The MiniPOV 4 Kit is a great 'next step' introduction to soldering. And once completed, you'll have another interactive toy to enjoy. Many years ago, about 20, I think, the first kit I soldered together was, in fact, the MiniPOV by Adafruit. But that original version was only red LEDs and had a serial port for programming. Oh, those were the days!

The MiniPOV 4 is a much different kit than the original. It features full-colour RGB LEDs and a USB port for programming, and they have even developed a program to allow you to easily design simple colour graphics and text to wave in mid-air, and display like magic. Beginners and advanced DIYers will love this kit. Check out the website for more info and a great tutorial on how to build your own!



VERDICT

MiniPOV 4 Kit

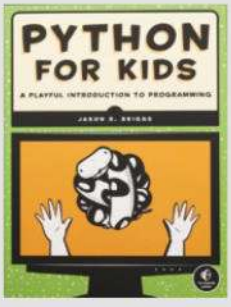
Fun to build; fun to use!

10/10

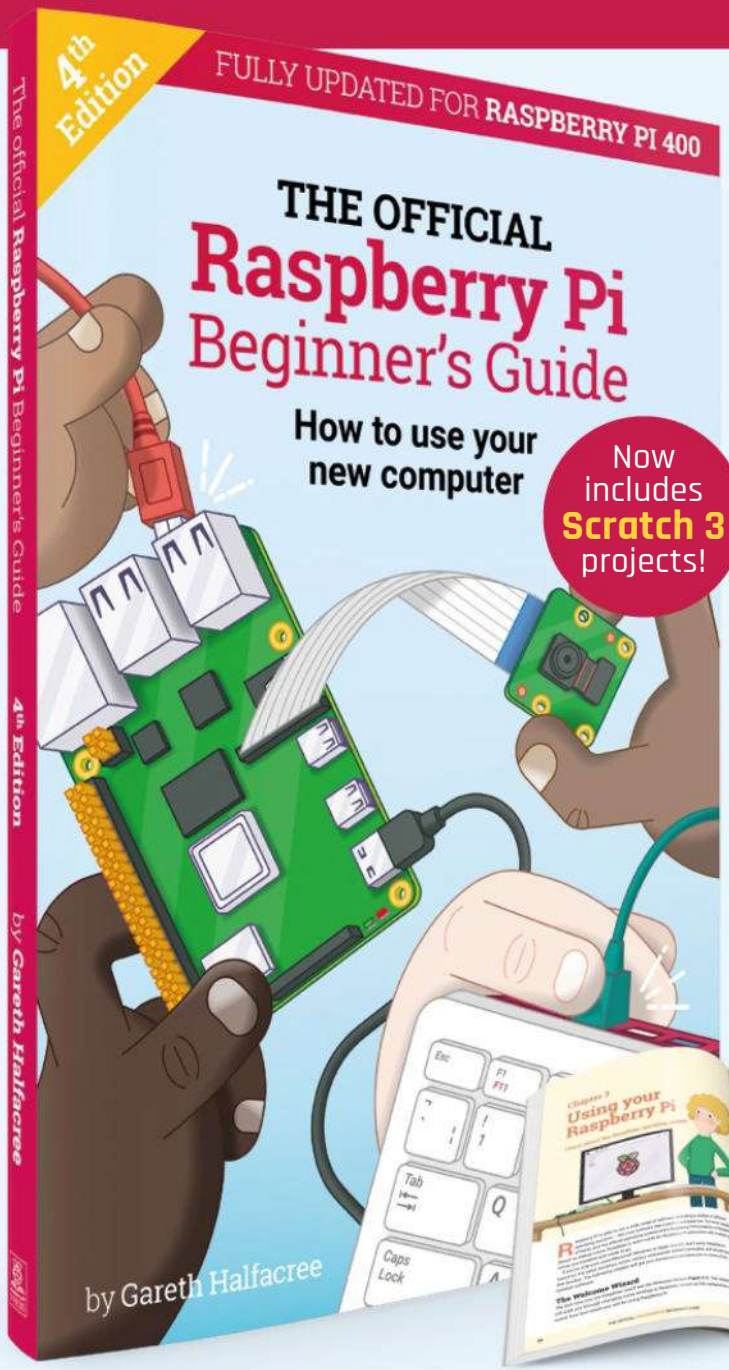
PYTHON FOR KIDS

NO STARCH | $34.95 | adafruit.com

Python is a great first language to learn. So many of the microcontrollers and single-board computers use Python as the main programming language. The *Python for Kids* book by Jason R. Briggs makes learning Python fun! You will learn the basics like data structures, control structures, and even build simple games and animations. If you are looking to learn code, Python is a very popular choice.



THE OFFICIAL Raspberry Pi Beginner's Guide



The only guide you need to get started with Raspberry Pi

Inside:

- Learn how to set up your Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch 3 and Python languages
- Create amazing projects by connecting electronic components to Raspberry Pi's GPIO pins

Plus much, much more!

Just £10

Buy online: magpi.cc/BGbook


FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
NEW 2022 UPDATE
RETRO GAMING WITH RASPBERRY PI
2ND EDITION
164 PAGES OF VIDEO GAME PROJECTS
+ BUILD AN ARCADE MACHINE
PLAY & CODE GAMES!

REVIEW

Pipistrelle

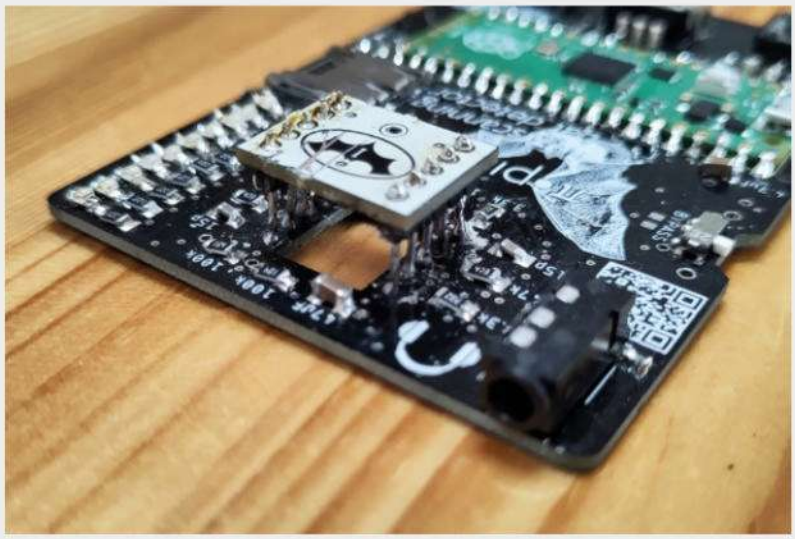
Build your own bat detector

£ varies | omenie.com/pipistrelles.html

By Ben Everard @ben_everard

The pipistrelle is a type of bat, and the Pipistrelle (sometimes styled π•pistrelle) is a tool for listening to ultrasonic sounds.

Obviously, it's one thing to add an ultrasonic microphone to a project, but you still need to do something with the signal it generates in order to make it audible to humans. Traditionally, most bat detectors are what's known as 'heterodyne'. In this style of detector, the ultrasonic signal from the bat is combined with another frequency that's very close. The two signals interact and make a signal that's the difference between the two. This generates a sound that's sort of like a series of pops and clicks that you may well have heard associated with bats. It's a reliable way of hearing bats, but what you're hearing isn't the actual sound of the bat, it's a heavily processed signal.



Below
We had a problem here where we had to desolder the daughterboard to flip the op-amp chip around. It's ended up looking messy, but it worked

The Pipistrelle can work in heterodyne mode, and also in time-expanding mode. In this latter mode, it slows down the signal to put the ultrasonic signals received from the microphone into the audio range. This way, you can hear the bat calls as they actually sound, which is completely unlike the sounds from heterodyne bat detectors.

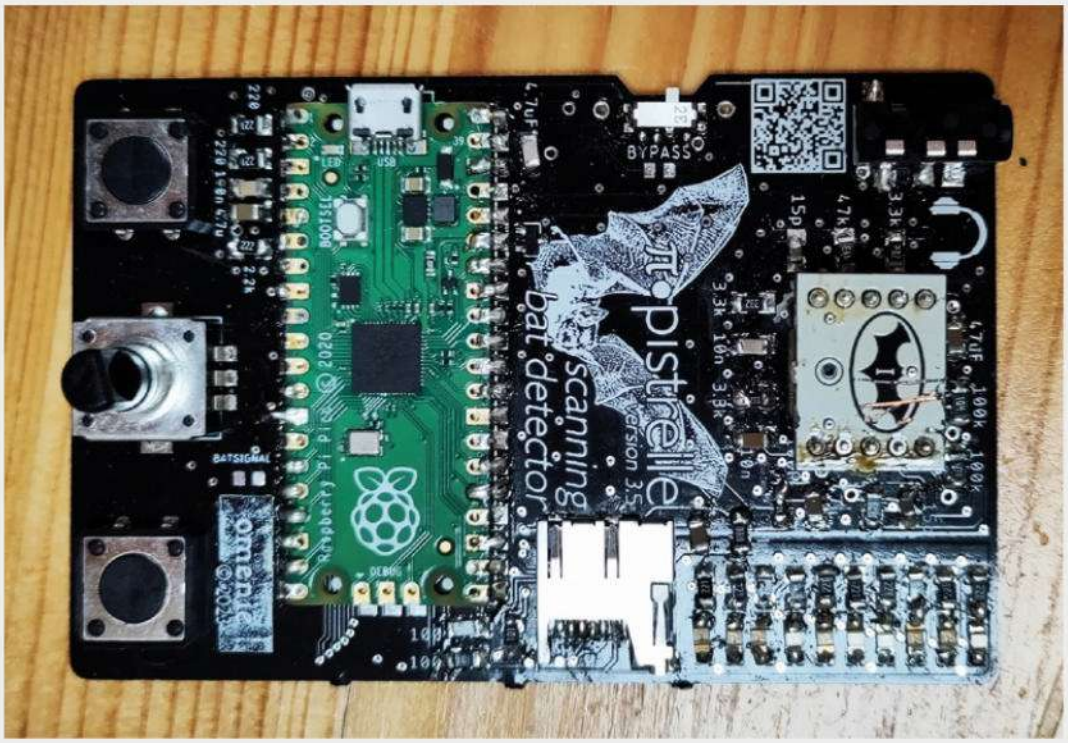
This ability to use both ways of listening to bats makes the Pipistrelle a great product for listening to bats, and one that doesn't really have an equivalent at even close to the price range (about £20 per device if you build a few).

While the product is great to use, it doesn't come as a product. In fact, it doesn't come as anything – it's just a design and bill of materials. You have to get the PCBs manufactured and source the parts yourself. The board designs aren't available, but are uploaded to PCBWay, so you can get them manufactured from there, but you can't use a different fabricator.

The parts are mostly surface-mount, and it's recommended that you get a solder stencil and solder it with a hotplate. We didn't do this because the cost of the solder stencil would nearly have doubled the cost of our single device. If you're soldering more, then the single stencil can be used multiple times, and it's probably a very sensible investment.

Most of the board is fairly straightforward to hand-solder. The only exceptions are the SD card slot which needs a pointy soldering iron and a steady hand, and the microphone, which is all but impossible to solder with an iron. We placed solder paste by hand and used a hotplate. We were successful, but it's a risky solder because if any paste gets into the microphone hole, it'll stop working. It would have been an easier and quicker build with a stencil, but that comes at a cost.

At this point, we should mention that there's been a new version since we tested it out, and the version we



Left
Once it's been built, Pipistrelle is a great product for listening into a usually inaudible world

used had the microphone on a separate board, so it was easy to solder separately. The new version has the microphone on the main board, so you'll have to solder it first if you want to go down the hand-assembly route.

The biggest problem we had was the lack of documentation. It was a bit tricky to work out how everything went together and how to get it working. The situation has improved since then, but do have a look through before you start, and make sure you're happy with the level of documentation before committing to a build.

> Do have a look through before you start, and make sure you're **happy with the level of documentation**

This is a really interesting product, and it's a hard one to review, because should we review it based on the finished product or the build? It's fun to play with, and the equivalent commercial product is expensive. However, this doesn't mean it's a toy. It can help identify bats and record their calls for later playback – it's certainly a useful device for anyone interested in flying mammals. However, it's a tricky build, and debugging problems is difficult. Unless you're experienced with more complex PCB construction, it's going to be a challenging build.

Whether or not 'challenging' is a good thing or not depends on what you like. For some people, this will be a fun way to push their skills. For others, it'll be frustrating and annoying. Hopefully, you know which camp you fall into.

Because of the cost structure of the parts, it only really becomes cost-effective once you're making a small batch (around five or ten). That's not to say you can't make one – we did – but to avoid the cost of the stencil, the soldering is a bit tricky.

There are certainly people for whom the Pipistrelle is a great product, but it's not without limitations. Most of these are down to the hand assembly and, in small-batch manufacture (such as using pick-and-place machines rather than hand assembly), would go away.

Left
This is a fiddly, but not impossible, component to hand-solder

VERDICT

A powerful ultrasonic detector, but difficult to make.

8/10

REVIEW

Arduino Nano ESP32-S3

Arduino takes its first steps into MicroPython

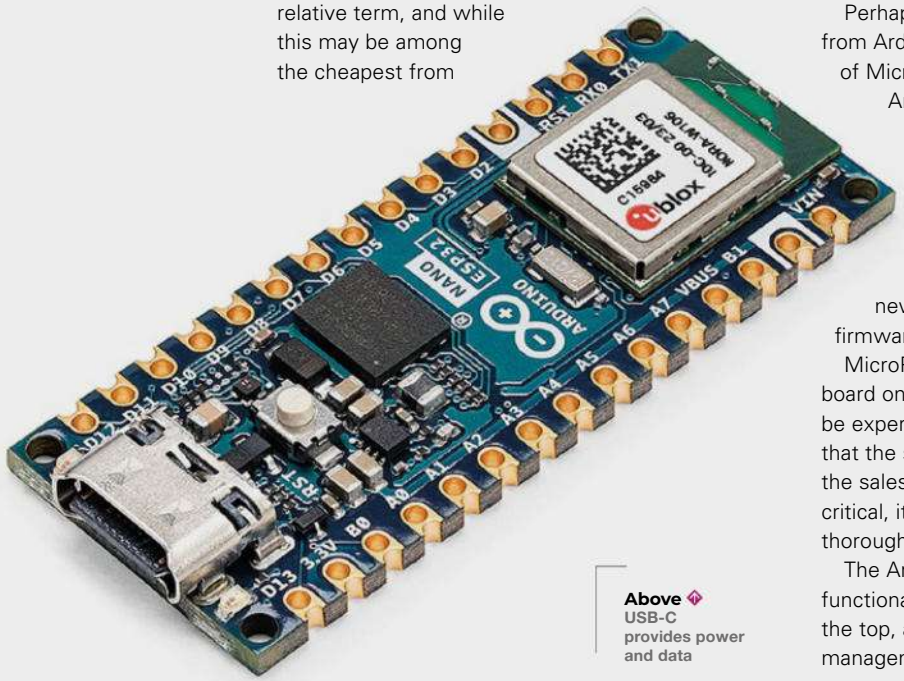
ARDUINO ◆ €18 | hsmag.cc/NanoESP32

By Ben Everard @ben_everard

If you look at Arduino's Nano line, you'll see that many options use a u-blox module for network access. However, the Arduino Nano ESP32-S3 is unusual because it doesn't have another controller.

The W106 Wi-Fi module contains an ESP32-S3 microcontroller with a dual-core processor that can run at up to 240MHz. This is beefy enough to run sketches alongside its connectivity duties.

By stripping it back to a single microcontroller, this board is the cheapest Wi-Fi-connected board in the Arduino range, at 18 euros. In fact, only one Arduino Nano is cheaper – the Nano Every. Cheapness is a relative term, and while this may be among the cheapest from Arduino, ESP32 boards are available from other manufacturers for significantly less.



Above ◆ USB-C provides power and data

Of course, not all ESP32 boards are created equal. This is a solidly made board that can be powered by a wide range of voltages. The Nano pinout gives access to 14 digital I/O pins (of which five can be PWM) and eight analogue inputs. There are two UARTs and one each of SPI and I2C. There's 512KB of SRAM, and a chunky 16MB of flash for your programs and data. This comes in the standard Nano form factor. While this isn't the most common form factor around, there is an ecosystem of parts that this board should work with (though do check the documentation for exact requirements).

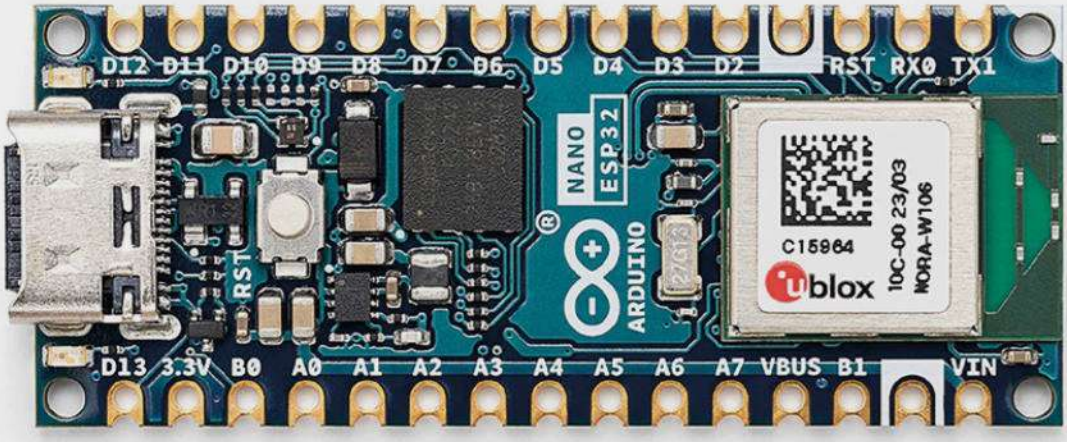
Perhaps the most unusual thing about this board from Arduino is that it comes with an official port of MicroPython (it can also be programmed in Arduino C++ using the traditional IDE).

While Arduino has been talking about the Python programming language for a long time, this is the first time it's been available on a board in a way that's both official and easy to use.

The software is supported through some new Arduino software – there's a MicroPython firmware flasher and a MicroPython IDE.

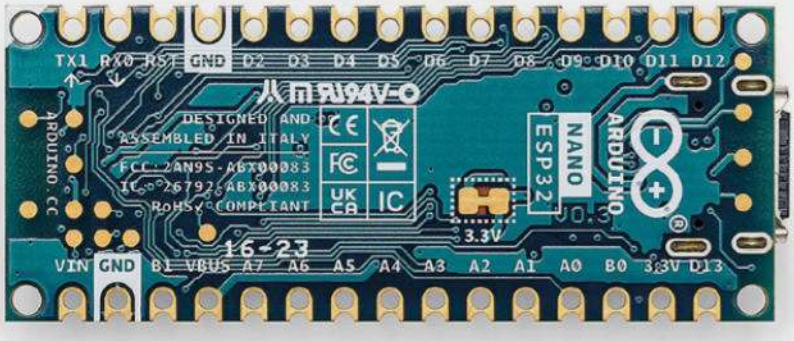
MicroPython support is listed as a feature of the board on the web page; the software itself claims to be experimental and pre-release. It's a bit frustrating that the status of the software isn't highlighted on the sales page, and if you're working on something critical, it's probably worth avoiding until it's more thoroughly tested.

The Arduino MicroPython editor is basic but functional. There's a toolbar of circular buttons at the top, a text editor, and optional panes for file management, and a terminal – it will look familiar to



Left
There's a lot squeezed onto this little board

Below
The flat bottom means that this can be soldered down into a project



anyone who's used the Mu Python editor. It gives you the basics of what you need for MicroPython. This editor will suit beginners more than experienced programmers. However, you don't need to use this editor – you can connect to the board with any other MicroPython IDE, such as Thonny, if you prefer.

Arduino has released a free MicroPython 101 web course which consists of a series of lessons that take you through getting started with this language. It skips quite quickly over the basics of programming, so if you've not coded before, you might struggle. Otherwise, it does a good job of introducing the language. However, there's one big caveat. It relies on specific hardware. For example, the button doesn't use a regular button – it uses a button module. The same with LEDs, potentiometers, etc. The code is the same either way, but if you don't know how to wire up, for example, an LED and series resistor, you won't know what to do. This reliance on specific hardware adds a lot to the cost of following the course.

As well as traditional Arduino C++ and MicroPython, you can use the Arduino IoT Cloud, an online service for working with devices remotely. This allows you to share data between devices, and even use machine learning to gain additional insights from the data your microcontrollers detect. This is a huge platform – far too big to accurately summarise here – and we'll take a detailed look in an upcoming issue. This online system is available on a range of network-enabled microcontroller boards from both Arduino and third-party manufacturers.

This board is perhaps the most unusual Arduino board we've reviewed. It's the first step into MicroPython, which is something we welcome. While Arduino C++ has defined a generation of maker electronics, it's no longer the best route into microcontroller programming for beginners, in our opinion. While Arduinos were the first widely-used hobbyist microcontroller boards (and the organisation used that first-mover advantage to cement its place in the market), it's a late-comer to the world of MicroPython. Many other options are both cheaper and more mature. The only real selling points of this board are its robustness and that it comes in the Nano form factor – if you already have hardware set up for this that you want to use with MicroPython, then it could be a reasonable option.

None of this is to say that the Nano ESP32-S3 is a bad board, but it does feel a few steps behind the market. If Arduino wants to be relevant in the world of embedded Python, it need something more than a well-made board with official support – there are already loads of these from a variety of manufacturers. The documentation could have been the reason to recommend this board, but the choice of expensive hardware makes this a less attractive proposition.

We'd love another player to come in and help push the MicroPython experience forward, particularly for new users, but at the moment, the Arduino Nano ESP32-S3 doesn't stand out. ◻

VERDICT

A perfectly capable board, but it struggles to justify its price tag.

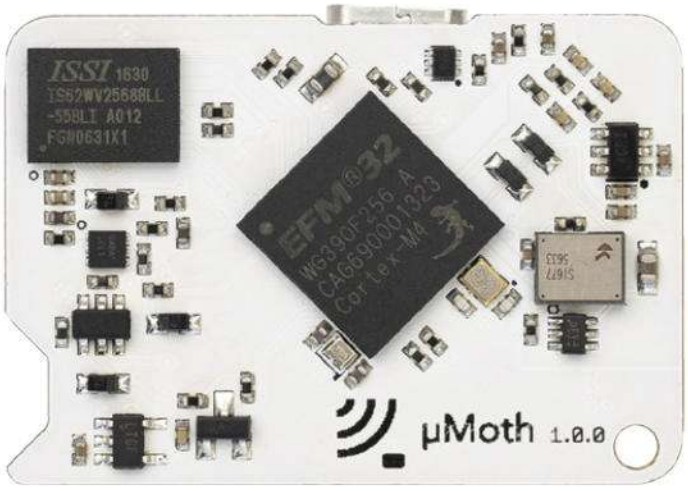
8/10

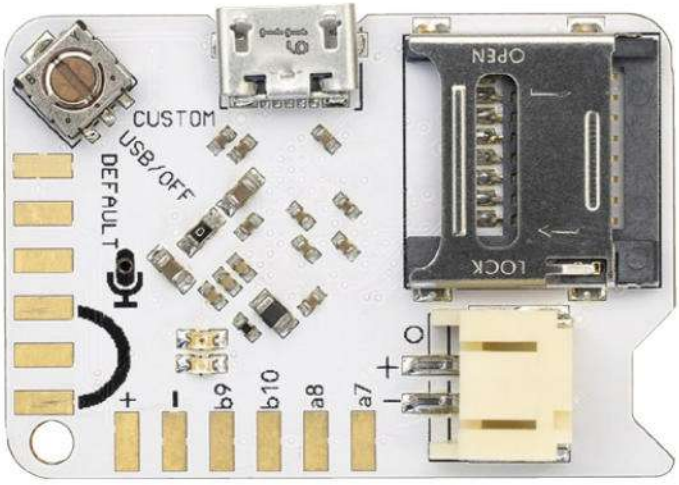
CROWDFUNDING NOW

µMoth

Listen in on the natural world

From $99 | **groupgets.com** | **Delivery: September 2023**





Microphones don't perfectly capture audio. They are tuned to their specific function. A laptop microphone might be tuned to someone speaking. Musicians have different microphones for different purposes.

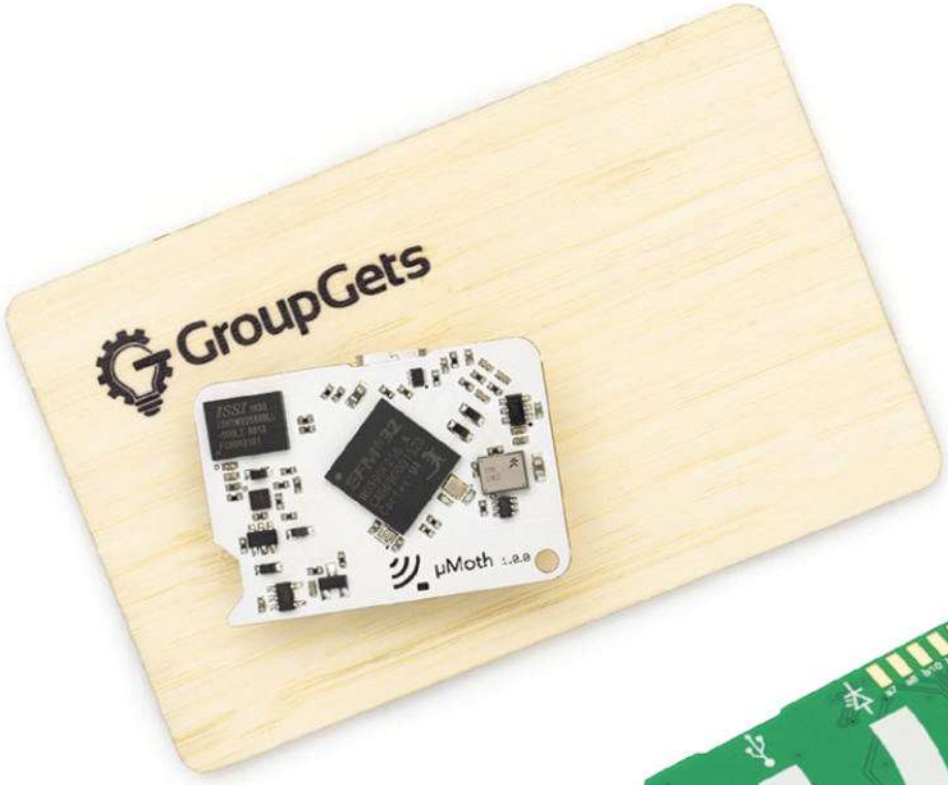
AudioMoth has been a popular device for picking up a wide range of audio frequencies, both those within human hearing and ultrasonic ranges. It can either save these as WAV files or stream them directly to a PC (much like a regular USB microphone).

The µMoth (pronounced 'micro-moth') is the same hardware as the original AudioMoth but shrunk down. At just 25×36mm and 4g in weight, it should be possible to jam this just about anywhere you want to capture audio. In most cases, this is going to be some kind of environmental monitor, but this could go anywhere you want to capture a wide range of audio input. It could, for example, be used to monitor mechanical hardware.

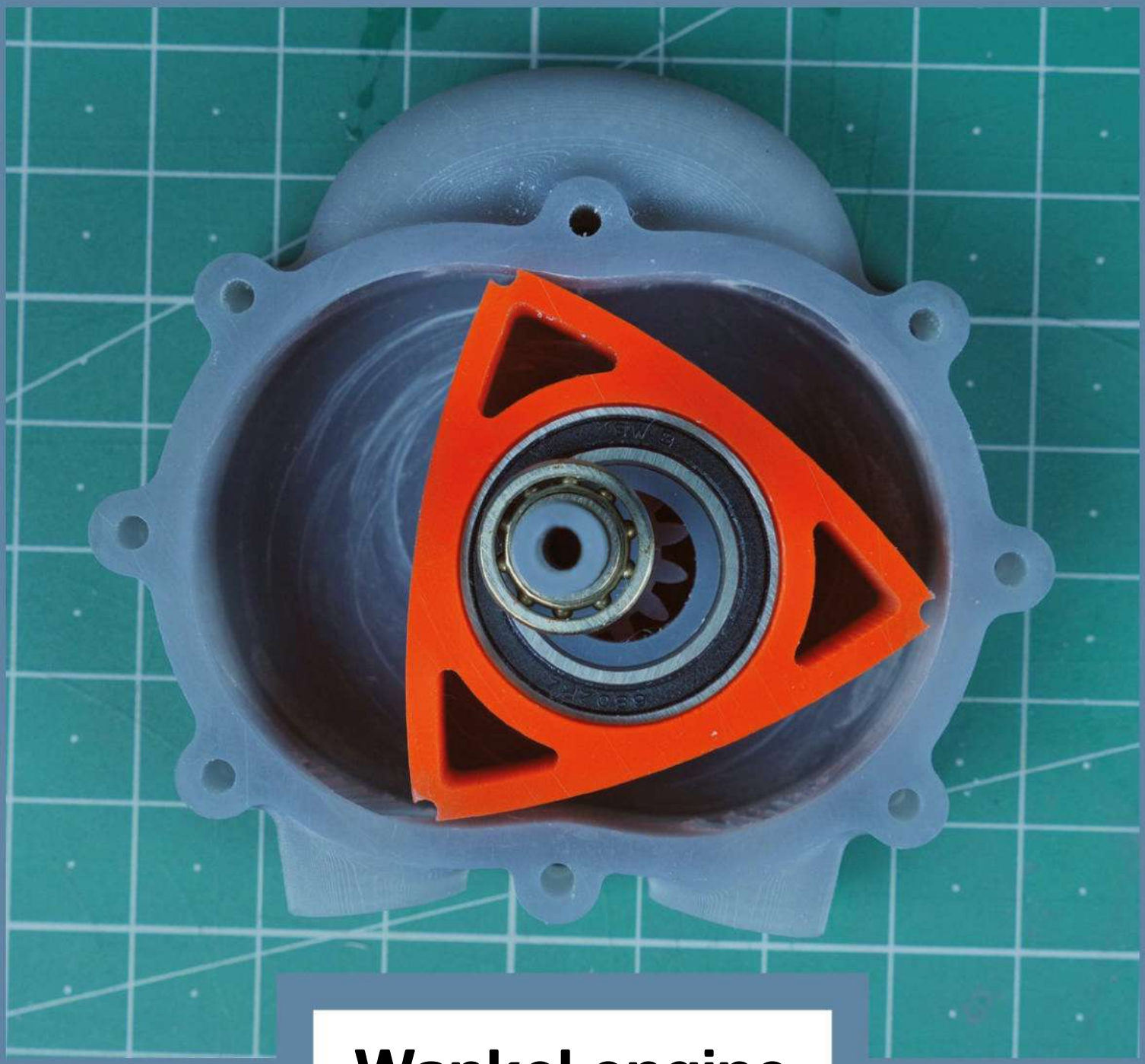
AudioMoth has been a great and affordable tool for scientists, so it's exciting to see how this smaller version will be used to learn more about the world around us.

BUYER BEWARE

When backing a crowdfunding campaign, you are not purchasing a finished product, but supporting a project working on something new. There is a very real chance that the product will never ship and you'll lose your money. It's a great way to support projects you like and get some cheap hardware in the process, but if you use it purely as a chance to snag cheap stuff, you may find that you get burned.



> **AudioMoth has been a popular** device for picking up a wide range of audio frequencies



Wankel engine

3D printing is the gift that keeps on giving. The internet gives us access to all sorts of patent diagrams, so if you have a modicum of 3D modelling ability, you can transfer century-old inventions into a computer and, from there, you can make it real in plastic (or even 3D-printed in metal).

This Wankel engine, printed by Joel Gomes, was a pretty cool toy when he first made it; with a bit of compressed air you could probably make it work. But now that the price of metal 3D printing has come down so much, putting it within reach of home engineers, it could be a stepping stone to a working combustion engine. We're living in a golden age!

TRUST
STARTS HERE

From genuine, manufacturer-warrantied components to millions of in-stock parts shipped same day, be confident Digi-Key will get you what you need—when you need it.

Visit **digikey.co.uk** today, or call 0800 587 0991.

Digi-Key ELECTRONICS